PENGENALAN CITRA TULISAN TANGAN DENGAN METODE *BACKPROPAGATION*

Skripsi

Disusun untuk melengkapi syarat-syarat

guna memperoleh gelar Sarjana Sains

ALPHIEN ANDANA

3125120206

PROGRAM STUDI MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS NEGERI JAKARTA

2017

LEMBAR PERSETUJUAN HASIL SIDANG SKRIPSI

PENGENALAN CITRA TULISAN TANGAN DENGAN METODE *BACKPROPAGATION*

Nama : Alphien Andana

No. Registrasi : 3125120206

	Nama	Tanda Tangan	Tanggal
Penanggung Jawab			
Dekan	: Prof. Dr. Suyono, M.Si.	.....	.....
	NIP. 19671218 199303 1 005		
Wakil Penanggung Jawab			
Pembantu Dekan I	: Dr. Muktiningsih, M.Si.	.....	.....
	NIP. 19640511 198903 2 001		
Ketua	: Drs. Mulyono, M.Kom.	.....	.....
	NIP. 19660517 199403 1 003		
Sekretaris	: Dr. Lukita Ambarwati, S.Pd, M.Si.	.....	.....
	NIP. 19721026 200112 2 001		
Penguji	: Ir. Fariani Hermin, M.T.	.....	.....
	NIP. 19600211 198703 2 001		
Pembimbing I	: Ratna Widyati, S.Si, M.Kom.	.....	.....
	NIP. 19750925 200212 2 002		
Pembimbing II	: Med Irzal, M.Kom.	.....	.....
	NIP. 19770615 200312 1 001		

Dinyatakan lulus ujian skripsi tanggal: 24 Januari 2017

ABSTRACT

ALPHIEN ANDANA, 3125120206. Handwriting Image Recognition Using Backpropagation Method. Thesis. Faculty of Mathematics and Natural Science Jakarta State University. 2017.

Backpropagation is one method of the neural network which is excellent in dealing with the introducing of complex pattern, one of which is the image of handwriting recognition. Issues raised in this research is the manufacturing system using a network of backpropagation method for image recognition handwriting, problem analysis process is carried out by determining the problem, literature, collecting research data, design the system, making the system and test the system. From the test result obtained by the level of system accuracy of the system in recognizing handwriting image with backpropagation method 1s 96%. Network architecture used is the variation of the number of iteration is 22, 0.05 learning rate, and the number of neuron hidden layer is 40. For the cases discussed, the network architecture produces mean square error (MSE) of $4.48e^{-14}$ *with 123 seconds training time.*

Keywords *: backpropagation, neural network, artificial intelligence, handwriting image, image recognition, GUI matlab, image recognition application.*

ABSTRAK

ALPHIEN ANDANA, 3125120206. Pengenalan Citra Tulisan Tangan dengan Metode *Backpropagation*. Skripsi. Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Jakarta. 2017.

Backpropagation merupakan salah satu metode pada jaringan syaraf tiruan yang sangat baik dalam menangani masalah pengenalan pola-pola kompleks, salah satunya adalah pengenalan citra tulisan tangan. Permasalahan yang diangkat dalam penelitian ini adalah pembuatan sistem dengan menggunakan jaringan metode *backpropagation* untuk pengenalan citra tulisan tangan, proses analisis masalah yang dilakukan adalah dengan menentukan masalah, studi pustaka, mengumpulkan data-data penelitian, merancang sistem, membuat sistem dan menguji sistem. Dari hasil pengujian sistem diperoleh tingkat akurasi sistem dalam mengenali citra tulisan tangan dengan metode *Backpropagation* adalah sebesar 96%. Arsitektur jaringan yang digunakan adalah dengan variasi jumlah iterasi 22, *learning rate* 0,05 dan jumlah *neuron hidden layer* 40. Untuk kasus yang dibahas, arsitektur jaringan tersebut menghasilkan Mean Square Error (MSE) sebesar $4.48e^{-14}$ dengan waktu *training* 123 detik.

Kata kunci : *backpropagation*, jaringan syaraf tiruan, kecerdasan buatan, citra tulisan tangan, pengenal citra, GUI matlab, aplikasi pengenalan citra.

PERSEMBAHANKU...

"Sesungguhnya orang orang yang berkata : "Tuhan kami ialah Allah" kemudian mereka meneguhkan pendirian mereka, maka malaikat malaikat akan turun pada mereka (dengan berkata) : "Janganlah kamu merasa takut; dan janganlah kamu bersedih hati; dan bergembiralah kamu dengan (memperoleh) surga yang telah dijanjikan kepadamu." "

-Q.S Fushshilat:30

Skripsi ini kupersembahkan untuk Papa, Mama, Chyka, Iqbal dan Ara.

"Terima kasih atas dukungan, do'a, serta kasih sayang kalian".

KATA PENGANTAR

Segala puji dan syukur kepada Allah SWT, yang telah melimpahkan rahmat, nikmat dan karunianya, sehingga penulis dapat menyelesaikan skripsi yang berjudul "Pengenalan Citra Tulisan Tangan dengan Metode *Backpropagation*" yang merupakan salah satu syarat dalam memperoleh gelar Sarjana Jurusan Matematika Universitas Negeri Jakarta.

Skripsi ini berhasil diselesaikan dengan adanya bantuan dari berbagai pihak. Oleh karena itu, dalam kesempatan ini penulis ingin menyampaikan terima kasih terutama kepada:

1. Ibu Ratna Widyati, S.Si., M.Kom selaku Dosen Pembimbing I dan Bapak Med Irzal, M.Kom. selaku Dosen Pembimbing II, yang telah meluangkan waktunya dalam memberikan bimbingan, bantuan, saran, nasehat serta arahan sehingga skripsi ini dapat menjadi lebih baik, terarah dan dapat diselesaikan.

2. Ibu Dr. Lukita Ambarwati. S,Pd., M.Si., selaku Ketua Program Studi Matematika FMIPA UNJ yang telah banyak membantu penulis.

3. Ibu Ir. Fariani Hermin Indiyah, M.T., selaku Pembimbing Akademik atas segala bimbingan dan kerja sama Ibu selama perkuliahan, dan seluruh Bapak/Ibu dosen atas pengajarannya yang telah diberikan, serta karyawan/karyawati FMIPA UNJ yang telah memberikan informasi yang penulis butuhkan dalam menyelesaikan skripsi.

4. Papa dan Mama yang selalu memberikan motivasi, semangat, dukungan, doa, serta cinta dan kasih sayang yang tulus kepada penulis.

5. Kedua adik perempuan dan satu adik laki-laki penulis, Chyka, Ara, dan

Iqbal, serta Kak arya yang terus memberi semangat, motivasi, mendoakan, menghibur, serta mendengar keluh kesah penulis ketika penulis mengalami kesulitan dalam penulisan skripsi.

6. Teman Cantik, Zuhai, Uyun, Bety, Mira, Dwi, Aan, Dewanti, Fatma, Yohana, serta semua teman-teman matmur 2012 UNJ, yang selalu setia menemani dan membantu penulis selama masa perkuliahan di Matematika UNJ.

7. Teman sejak muda, Vinny, Yudho, Desty, Ucan, Moci dan Ulfa, yang selalu menghibur dengan candaan yang memotivasi agar penulis segera menyelesaikan skripsi.

8. RPL 3rd, untuk segala tawa dan semangat yang sangat membantu dan menghibur penulis, serta untuk senantiasa berbagi pengalaman dan informasi dalam banyak hal.

9. Kakak angkat di matematika 2011, Kak Danti yang telah memberi motivasi, informasi, saran, dan membantu penulis selama menjalani masa perkuliahan.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna. Masukan dan kritikan akan sangat berarti. Semoga skripsi ini dapat bermanfaat bagi pembaca sekalian.

Jakarta, Januari 2017

Alphien Andana

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

Pada bab ini akan dibahas mengenai latar belakang masalah, perumusan masalah, pembatasan masalah, tujuan penulisan, manfaat penulisan, dan metode penilitian.

1.1 Latar Belakang Masalah

Pesatnya perkembangan zaman menuntut manusia untuk berfikir kreatif dan inovatif dalam mengembangkan ilmu pengetahuan dalam berbagai bidang guna membantu serta mempermudah pekerjaan manusia. Salah satu objek pengembangan teknologi yang cukup pesat saat ini adalah komputer. Pengembangan teknologi komputer ini dilakukan pada perangkat keras *(hardware)* serta perangkat lunak *(software)* yang dapat menirukan kecerdasan manusia (kecerdasan buatan).

Salah satu teknik menirukan kecerdasan manusia adalah teknik pengenalan pola *(pattern recognition)* dengan metode jaringan syaraf tiruan. Permasalahan yang dimiliki oleh sistem komputer dalam pengenalan pola ini adalah bagaimana caranya supaya sistem komputer dapat mengenali berbagai macam tulisan tangan angka dan karakter dari berbagai input yang ditulis manual dapat diotomatisasi ke dalam sistem komputer untuk mempermudah dalam pencatatan dokumen-dokumen penting pada perusahaan atau lembaga perkantoran. Beberapa diantaranya adalah pada lembaga kantor pos untuk pengenalan tulisan alamat pada surat yang ditulis manual, pengenalan tulisan

nomer rekening pada slip bukti pembayaran atau setoran pada bank, pengenalan tulisan nomer polisi pada kendaraan bermotor, dan pada aplikasi-aplikasi lainnya. Pengenalan tulisan tangan ini juga bermanfaat untuk piranti elektronik interaktif seperti pada *tablet Personal Computer*(PC), dan lain sebagainya.

Agar sistem mampu mengenali pola citra tulisan tangan manusia, sebelumnya harus diambil informasi yang mewakili citra tersebut atau biasa disebut dengan data input, data informasi tentunya harus berupa data digital. cara untuk memperoleh data citra digital adalah dengan melakukan proses *scan* pada gambar. Dari hasil *scan* akan diperoleh citra yang kemudian dilakukan operasi *preprocessing*. Setelah dilakukan proses tersebut, maka dapat dibuat sebuah sistem cerdas pada komputer yang mampu mengenali tulisan tangan. Pada Jaringan syaraf tiruan ada beberapa metode atau algoritma yang dapat digunakan untuk mengenali suatu tipe pola khususnya dalam pengenalan pola tulisan tangan, salah satunya adalah *Backpropagation*.

Backpropagation adalah metode yang cukup baik dalam menyelesaikan masalah pengenalan pola-pola kompleks. Metode *Backpropagation* merupakan salah satu metode jaringan saraf tiruan yang populer. Istilah *Backpropagation* didasari oleh cara kerja jaringan ini, yaitu bahwa *error* dari unit tersembunyi (*Hidden layer*) diturunkan dari penyiaran kembali *error* yang dihubungkan dengan unit output. "*Backpropagation* melatih jaringan untuk mendapatkan keseimbangan antara kemampuan jaringan untuk mengenali pola yang digunakan selama pelatihan serta kemapuan jaringan untuk memberikan respon yang benar terhadap pola masukan yang serupa (tetapi tidak sama) dengan pola yang dipakai selama pelatihan" (Siang, 2005:97). Metode *Backpropagation* menggunakan 2 alur dalam perhitungan bobot, yaitu propagasi maju (*forward*) dan propagasi mundur (*backward*).

Penelitian-penelitian tentang aplikasi penggunaan jaringan syaraf tiruan

melalui pengenalan pola telah banyak dikaji dalam berbagai bidang ilmu pengetahuan. Penelitian yang dilakukan sebelumnya adalah "Aplikasi Jaringan Syaraf Tiruan dengan Menggunakan Metode *Backpropagation* dalam Kasus Pengenalan Pola Huruf Hijaiah" (Anggi, 2013). Penelitian lain yang berhubungan dengan pengenalan pola adalah yang dilakukan oleh M. Kayadoe, Dkk tahun 2013 yaitu "Sistem Pengenalan *Barcode* Menggunakan Jaringan Syaraf Tiruan *Backpropagation*".

Berdasarkan uraian diatas peneliti tertarik untuk melakukan penelitian mengenai **Pengenalan Citra Tulisan Tangan dengan Metode *Backpropagation***.

1.2 Perumusan Masalah

Berdasarkan latar belakang diatas dapat dirumuskan permasalahan yang akan dikaji adalah :

1. Bagaimana proses pengolahan citra tulisan tangan yang merupakan data input, agar kemudian dapat digunakan untuk pembuatan jaringan dan dapat dikenali oleh sistem?

2. Bagaimana cara menerapkan jaringan syaraf tiruan metode *backpropagation* dan teknik pengolahan citra dalam teknologi informasi untuk membuat sistem perangkat lunak yang mampu mengenali citra tulisan tangan?

3. Arsitektur jaringan yang seperti apa, yang cukup baik digunakan untuk pengenalan citra angka tulisan tangan?

4. Angka berapa saja yang paling mudah dan paling sulit dikenali oleh sistem?

1.3 Pembatasan Masalah

Pembatasan masalah dalam penulisan ini adalah:

1. Pengenalan tulisan tangan dilakukan data input berupa lembar kertas yang kemudian di-*scan* menggunakan *scanner*.
2. Penelitian ini hanya terbatas pada masalah pengenalan tulisan tangan yang terdiri dari angka tunggal 9 sampai 0, tidak menggabungkan jenis angka.
3. Penelitian menggunakan pendekatan studi kasus per individu untuk 10 (sepuluh) input gambar yang terdiri dari angka tunggal 9-0 kepada 10 (sepuluh) orang yang berbeda, sehingga diperoleh jumlah eksekusi huruf tulisan tangan yang diharapkan sebanyak 100 jumlah angka.
4. Arsitektur jaringan yang digunakan adalah jaringan layar jamak dengan satu buah *hidden layer*. Dengan jumlah *neuron hidden layer* yang digunakan adalah 40, maksimal epoch yang digunakan adalah 1000, serta *learning rate*yang digunakan adalah 0.05.
5. Fungsi aktivasi yang digunakan adalah fungsi sigmoid biner.
6. *software* yang digunakan untuk pembuatan program adalah Matlab R2012b.

1.4 Tujuan Penulisan

Tujuan yang ingin dicapai dalam skripsi ini adalah untuk dapat membuat sebuah perangkat lunak yang dapat mengenali citra tulisan tangan khususnya untuk angka sembilan (9) sampai nol (0) dengan menggunakan jaringan syaraf tiruan dengan metode *Backpropagation*.

1.5 Manfaat Penulisan

Manfaat yang diharapkan dari skripsi ini adalah

1. Bagi peneliti :

   Hasil penelitian ini diharapkan dapat menambah wawasan dan pengetahuan serta menjawab keingin tahuan peneliti mengenai jaringan syaraf tiruan metode *Backpropagation* dalam kasus pengenalan citra tulisan tangan. Diharapkan pula bagi peneliti untuk dapat mengetahui dan memahami proses pembuatan sistem perangkat lunak untuk pengenalan citra tulisan tangan pada *software* Matlab R2012b.

2. Bagi pembaca :

   Hasil penelitian ini diharapkan dapat digunakan sebagai bahan perbandingan bagi pihak lain yang ingin melakukan penelitian lebih lanjut serta dapat digunakan sebagai refrensi untuk pengembangan penelitian berikutnya. Diharapkan pula hasil penelitian ini dapat bermanfaat untuk menambah wawasan dan pengetahuan pembaca.

1.6 Metode Penelitian

Skripsi ini merupakan rekayasa produk dalam bidang matematika yang didasarkan pada buku-buku dan jurnal-jurnal tentang teori permasalahan di bidang matematika komputasi khususnya jaringan syaraf tiruan metode *Backpropagation*.

BAB II

LANDASAN TEORI

Pada bab ini akan dijelaskan mengenai dasar teori yang digunakan dalam penelitian, perancangan dan implementasi. Adapun dasar teori yang digunakan pada skripsi ini adalah mengenai matriks, pengenalan Citra digital, jaringan syaraf tiruan, serta metode *Backpropagation.*

2.1 Matriks

2.1.1 Definisi Matriks

Sebuah matriks adalah susunan segi empat siku-siku dari bilangan-bilangan, bilangan-bilangan dalam susunan tersebut dinamakan *entri* dalam matriks (Anton, 1987:22-23). Ukuran (*ordo*) suatu matriks dijelaskan dengan menyatakan banyaknya baris (horizontal) dan banyaknya kolom (vertikal) yang terdapat dalam matriks tersebut.

Notasi nama matriks biasanya menggunakan huruf kapital. Jika F adalah sebuah matriks, maka digunakan f_{xy} untuk menyatakan entri atau elemen yang terdapat didalam baris x dan kolom y dari F, jadi matriks dengan ukuran m x n beserta entri-entrinya secara umum dapat ditulis dengan berikut :

$$F(m*n) = \begin{pmatrix} f_{11} & f_{12} & \cdots & f_{1n} \\ f_{21} & f_{22} & \cdots & f_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ f_{m1} & f_{m2} & \cdots & f_{mn} \end{pmatrix} atau [f_{xy}]_{m*n}$$

Sebuah matriks berukuran m x 1 disebut sebagai vektor kolom, sedangkan matriks berukuran 1 x n disebut sebagai vektor baris (johnson & Wichern, 2007 : 88)

2.1.2 Perkalian Matriks

Jika F adalah matriks m x n dan G matriks m x n, maka hasil kali FG adalah matriks m x n yang entri-entrinya ditentukan sebagai berikut. Untuk mencari entri dalam $baris - x$ dan $kolom - y$ dari FG, pilihlah $baris - x$ dari matriks F dan $kolom - y$ dari matriks G. Kalikanlah entri-entri yang bersesuaian dari baris dan kolom tersebut bersama-sama dan kemudian tambahkanlah hasil kali yang dihasilkan (Anton, 1988 : 25)

Contoh :

$$Diketahui : F = \begin{pmatrix} 1 & 3 & 4 \\ 3 & 2 & 5 \end{pmatrix}; G = \begin{pmatrix} 2 & 4 \\ 3 & 6 \\ 1 & 3 \end{pmatrix}$$

Tinjaulah perkalian matriks F dan G. karena matriks F adalah berukuran $2 * 3$ dan matriks G berukuran $3 * 2$ maka hasil kali FG adalah matriks berukuran $2 * 2$. perhitungannya adalah sebagai berikut :

$$\begin{pmatrix} [(1*2)+(3*3)+(4*1)] & [(1*4)+(3*6)+(4*3)] \\ [(3*2)+(2*3)+(5*1)] & [(3*4)+(2*6)+(5*3)] \end{pmatrix} \rightarrow AB \begin{pmatrix} 15 & 34 \\ 17 & 39 \end{pmatrix}$$

2.2 Rekayasa Citra Digital

2.2.1 Citra Digital

Citra digital dapat didefinisikan sebagai fungsi dua variabel, f(x,y), dimana x dan y adalah koordinat spasial dan nilai f(x,y) adalah intensitas citra pada

koordinat tersebut. Kombinasi dari tiga warna dasar, yaitu merah, hijau, dan biru (*Red, Green, Blue* - RGB) merupakan teknologi dasar untuk menciptakan dan menampilkan warna pada citra digital .

Sensor optik yang terdapat di dalam sistem pencitraan disusun sedemikian rupa sehingga terbentuk bidang dua dimensi (x, y). Besar intensitas yang diterima sensor di setiap titik (x, y) disimbolkan oleh f(x, y) dan besarnya tergantung pada intensitas yang mampu dipantulkan oleh objek. Ini berarti f(x, y) sebanding dengan energi yang dipancarkan oleh sumber cahaya. Konsekuensinya, besar intensitas f(x, y) tidak bolah nol dan harus berhingga, yaitu :

$$0 < f(x, y) < \infty \tag{2.1}$$

Fungsi $f(x, y)$ dapat dipisahkan menjadi dua komponen, yaitu :

1. Jumlah cahaya yang berasal dari sumbernya disimbolkan oleh $i(x, y)$, nilainya terletak pada selang 0 dan ∞,

2. Derajat kemampuan objek memantulkan cahaya disimbolkan oleh $r(x, y)$, nilainya terletak pada selang 0 dan 1

Besar $f(x, y)$ merupakan kombinasi perkalian dari keduanya,

$$f(x, y) = i(x, y) * r(x, y) \tag{2.2}$$

dimana

$$0 < i(x, y) < \infty \tag{2.3}$$

dan

$$0 < r(x, y) < 1 \tag{2.4}$$

Derajat keabuan (*gray level*) merupakan intensitas $f(x, y)$ di titik (x, y), yang dalam hal ini derajat keabuan bergerak dari hitam ke putih, sedangkan

citranya disebut citra skala keabuan (*grayscale image*). Derajat keabuan memiliki rentang nilai [0, 1], yang dalam hal ini intensitas 0 menyatakan hitam, nilai intensitas 1 menyatakan putih

Citra digital yang berukuran m x n biasanya dinyatakan dalam bentuk matriks yang berukuran m baris dan n kolom, sebagai berikut:

$$f(x,y) \approx \begin{pmatrix} f(0,0) & f(0,1) & ... & f(0,n-1) \\ f(1,0) & f(0,1) & ... & f(1,n-1) \\ ... & ... & ... & ... \\ f(m-1,0) & f(m-1,1) & ... & f(m-1,n-1) \end{pmatrix}$$

2.2.2 Pengolahan Citra (*Pre Processing)*

Pengolahan gambar atau pengolahan citra yang sering disebut *image processing*, merupakan suatu proses yang mengubah sebuah gambar menjadi gambar lain yang memiliki kualitas lebih baik untuk tujuan tertentu (Ardhianto, dkk, 2011). Sari (2010) menyatakan bahwa pengolahan citra adalah pemrosesan citra, khususnya dengan menggunakan komputer, menjadi citra yang kualitasnya lebih baik sehingga citra akan lebih mudah dikenali, khususnya pada kasus pengenalan pola. Menurut Pratiarso, dkk (2009), sesuai dengan perkembangan komputer itu sendiri, pengolahan citra mempunyai dua tujuan utama, yaitu sebagai berikut:

1. Memperbaiki kualitas citra, dimana citra yang dihasilkan dapat menampilkan informasi secara jelas.

2. Mengekstrasi informasi ciri yang menonjol pada suatu citra, hasilnya adalah informasi citra dimana manusia mendapatkan informasi ciri dari citra secara numerik.

Proses pengolahan citra diterapkan pada citra dengan tujuan untuk:

1. *Image Acquisition*

   Image Acquisition dilakukan dengan *cropping*, yaitu memotong gambar menjadi bagian yang diinginkan dengan ukuran *pixel* yang sama untuk setiap citra, agar data yang diperoleh menjadi data yang homogen.

2. *Image Segmentation*

   Operasi ini bertujuan untuk mengelompokkan citra menjadi beberapa *region* berdasarkan kriteria tertentu, serta untuk menemukan karakteristik yang dimiliki oleh citra.

   Proses ini dilakukan apabila data yang diperoleh berupa citra warna (RGB) yang mempunyai nilai pixel cukup besar sehingga pengolahan citra akan sulit. Untuk itulah perlu konversi citra dari RGB ke *grayscale*.

3. *Image Enhancement*

   Operasi ini bertujuan untuk meningkatkan kualitas citra. Agar mempermudah saat proses pengenalan pola, gambar yang telah dikonversi ke dalam bentuk *grayscale* harus dipertajam kualitas citranya. Proses yang digunakan untuk mempertajam citra salah satunya adalah dengan penggunaan filter.

4. *Morphology*

   memperkecil ukuran citra agar mudah diolah dalam komputer *(image compression)*. Operasi *morphology* bertujuan untuk memperbaiki bentuk objek agar dapat menghasilkan fitur-fitur yang lebih akurat ketika analisis dilakukan terhadap objek.

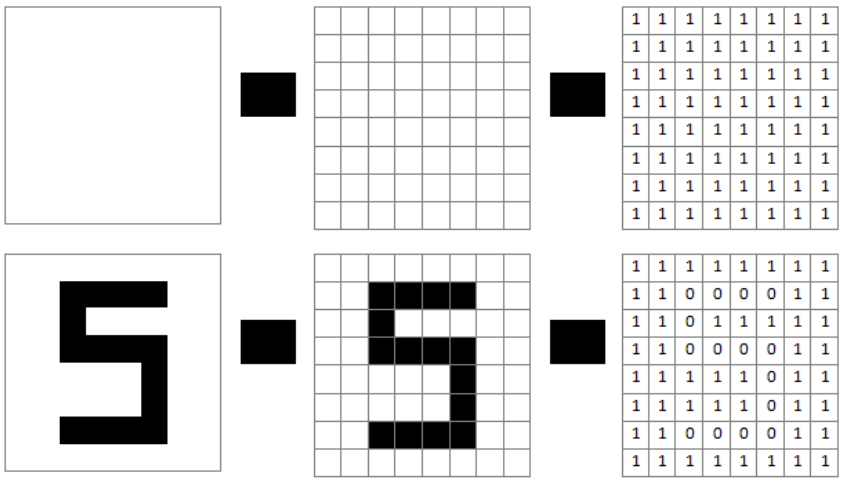
2.2.3 Pengenalan Citra

Pengenalan pola merupakan mengelompokkan secara otomatis oleh mesin (komputer) terhadap data numerik dan simbolik (termasuk citra). Pengelompokkan ini bertujuan untuk mengenali suatu objek di dalam citra. Otak manusia telah belajar untuk mengklasifikasi objek-objek di alam sehingga mampu mengenali objek yang dilihatnya serta membedakan suatu objek dengan objek lainnya. Kemampuan sistem visual manusia ini lah yang dicoba untuk ditiru oleh mesin.

Komputer memiliki cara pandang tersendiri terhadap suatu citra, biasa disebut sebagai *computer vision*. Citra pada komputer harus melalui beberapa tahapan yang cukup rumit, berbeda dengan proses fotografis seperti pada foto dapat dihasilkan suatu citra nyata yang langsung dapat dinikmati oleh indera penglihatan. Tahapan-tahapan tersebut dapat digambarkan sebagai suatu rangkaian proses dari proses akuisisi data, manipulasi data, visualisasi data, serta proses penyimpanan data.

Suatu citra harus direpresentasikan secara numerik dengan nilai-nilai diskrit agar dapat diolah dengan komputer. Digitalisasi merupakan representasi citra dari kontinu menjadi nilai-nilai diskrit. Citra yang dihasilkan dari proses representasi tersebut akan berupa citra digital. Citra digital akan berbentuk persegi panjang dengan dimensi ukuran yang dinyatakan sebagai tinggi x lebar.

Digitalisasi spasial (penerokan/sampling) merupakan salah satu proses digitalisasi citra adalah. Dalam proses digitalisasi spasial, sebuah citra kontinu diterok pada grid yang berbentuk bujursangkar.



Gambar 2.1: Digitali Spasial (Sampling)

2.2.4 Data Citra

Gambar-gambar yang selama ini dimiliki dan dilihat oleh manusia merupakan hasil pencitraan dari sebuah cahaya yang ditangkap oleh mata melalui retina (Gonzales, 2002).

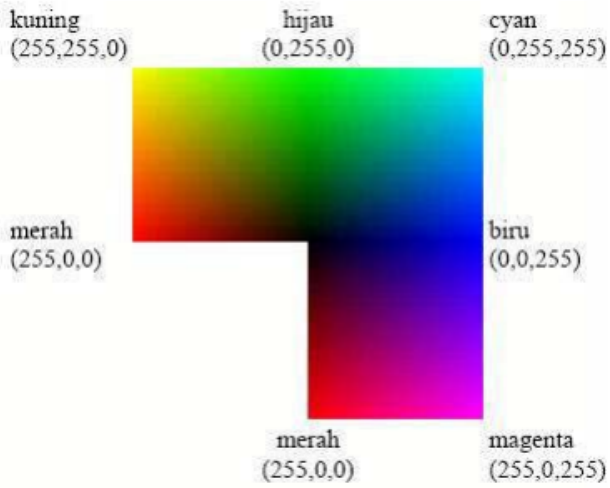
Menurut Sari (2010), komputer dapat mengolah isyarat-isyarat elektronik digital yang merupakan kumpulan sinyal biner (bernilai 0 dan 1). Oleh sebab itu, citra digital harus mempuyai format tertentu yang sesuai agar dapat merepresentasikan objek pencitraan dalam bentuk kombinasi data biner. Data citra gambar yang dapat diolah oleh komputer adalah berupa format *.JPEG, *BMP, *.TIF, *.GIF, *.PNG, dan *. FWD.

2.2.5 Pengolahan Warna

Pengolahan warna citra merupakan salah satu bagian penting sebelum dilakukannya pengenalan pada citra. Pengolahan warna ini dapat digunakan untuk mengekstrasi ciri citra guna memisahkan objek citra dengan *background*nya

Citra RGB

RGB adalah suatu model warna yang terdiri dari merah, hijau, dan biru, digabungkan dalam membentuk suatu susunan warna yang luas. Setiap warna dasar, merah, hijau, dan biru dapat diberi rentang-nilai. pada monitor komputer, nilai rentangnya paling kecil = 0 dan paling besar = 255. Sebuah jenis warna, dapat dibayangkan sebagai vektor di ruang 3 dimensi yang biasanya digunakan dalam matematika, koordinatnya dinyatakan dengan bentuk tiga bilangan, yaitu komponen-x, komponen-y, dan komponen-z. Misalkan sebuah vektor dituliskan sebagai r = (x,y,z). Untuk warna komponen-komponen tersebut digantikan oleh komponen *R(ed), G(reen), B(lue)*.



Gambar 2.2: Warna RGB

Pengolahan citra akan sulit dilakukan apabila citra berbentuk warna RGB. Untuk itulah perlu dibedakan intensitas dari masing-masing warna. Hal yang perlu dilakukan dalam proses ini adalah dengan melakukan pembacaan nilai-nilai R, G, dan B pada suatu pixel, menampilkan dan menafsirkan hasil perhitungan sehingga mempunyai arti sesuai yang diinginkan.

Salah satu cara yang mudah untuk menghitung nilai warna dan menafsir-

kan hasilnya dalam model warna RGB adalah dengan melakukan normalisasi terhadap ketiga komponen warna tersebut (Ahmad, 2005). Cara melakukan normalisasi adalah sebagai berikut:

$$r = \frac{R}{R+G+B} \tag{2.5}$$

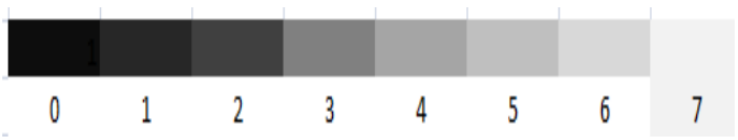
$$g = \frac{G}{R+G+B} \tag{2.6}$$

$$b = \frac{B}{R+G+B} \tag{2.7}$$

Nilai warna hasil normalisasi kemudian ditafsirkan dengan melihat besarannya. Jadi dominasi warna dapat dilihat dari besaran nilai tiap indeks. Warna primer (*Red, Green, and Blue*) merupakan tiga komponen citra yang direpresentasikan dalam model warna RGB.

Citra Gray

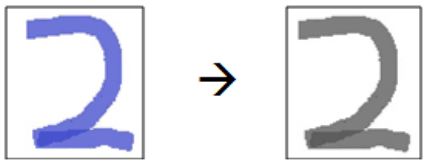
Citra *gray* merupakan citra digital yang hanya memiliki satu nilai kanal pada setiap pixelnya, sehingga nilai bagian warna *Red = Green = Blue*. Nilai tersebut digunakan untuk menunjukkan tingkat intensitas warnanya. Warna yang dimiliki oleh citra *gray* adalah warna dari hitam, keabuan, dan putih. Derajat keabuan disini merupakan warna abu dengan berbagai tingkatan dari hitam hingga putih. Citra *gray* memiliki derajat keabuan 8 bit seperti yang dapat dilihat berikut ini :



Gambar 2.3: Gradasi Warna *Grayscale*

Operasi *grayscale* bertujuan untuk merubah citra 24 bit RGB menjadi citra abu-abu. Pemrosesan pada tingkat abu-abu ini dipilih karena lebih sederhana dan hanya menggunakan sedikit kombinasi warna dan dengan citra abu-abu dirasakan sudah cukup untuk memproses suatu gambar. Perubahan citra 24 bit RGB menjadi citra abu-abu adalah dengan menghitung rata-rata dari intensitas:

$$0.299 * red, 0.587 * green, 0.114 * blue \tag{2.8}$$



Gambar 2.4: Citra RGB; Citra Gray

Grayscale sendiri merupakan sebuah proses pengolahan citra yang biasa dilakukan untuk mempertegas citra yang sebelumnya berupa citra warna lalu dirubah menjadi citra digital dengan skala keabuan.

Citra Biner

Citra biner diperoleh melalui proses pemisahan piksel-piksel berdasarkan derajat keabuan yang dimilikinya.

Gambar 2.5: Citra RGB; Citra Gray; Citra Biner

Gambar 2.5 memperlihatkan pixel yang memiliki derajat keabuan lebih

kecil dari nilai batas yang ditentukan akan diberikan nilai 0, sementara piksel yang memiliki derajat keabuan yang lebih besar dari batas akan diubah menjadi bernilai 1.

$$g(x,y) = \begin{cases} a_1,\ jika\ f(x,y) \leq T \\ a_2,\ jika\ f(x,y) > T \end{cases} \tag{2.9}$$

Jika $a_1 = 0$ dan $a_2 = 1$, maka operasi ini akan mentransformasikan suatu citra menjadi citra biner. Misal suatu citra memiliki gray level 256, dipetakan menjadi citra biner, maka fungsi trasformasinya adalalah sebagai berikut:

$$g(x,y) = \begin{cases} 0,\ jika\ f(x,y) \leq 128 \\ 1,\ jika\ f(x,y) > 128 \end{cases} \tag{2.10}$$

Pixel-pixel yang nilai intensitasnya di bawah 128 diubah menjadi hitam, sedangkan pixel-pixel yang nilai intensitasnya di atas 128 diubah menjadi putih.

Deteksi Tepi

Suatu objek yang berada dalam bidang citra dan tidak bersinggungan dengan batas bidang citra, itu berarti objek tersebut dikelilingi oleh daerah yang bukan objek, yaitu latar belakang. Tepi objek merupakan pertemuan antara bagian objek dengan bagian latar belakang. Tepi objek adalah salah satu bagian yang penting karena dapat mewakili informasi yang penting dari objek.

Tepi objek ditandai dengan adanya perubahan intensitas lokal yang terjadi di dalam citra. Tepi biasanya muncul pada batas antar dua daerah yang berbeda nilai intensitas dalam citra.

2.2.6 *Feature Extraction* (Ekstraksi Ciri)

Ciri merupakan karakteristik unik dari suatu objek. Ciri dibedakan menjadi dua yaitu ciri alami dan ciri buatan. Ciri alami merupakan bagian dari

gambar, misalnya kecerahan dan tepi objek. Sedangkan ciri buatan merupakan ciri yang diperoleh dengan operasi tertentu pada gambar, misalnya histogram tingkat keabuan. Sehingga ekstraksi ciri adalah proses untuk mendapatkan ciri-ciri pembeda yang membedakan suatu objek dari objek yang lain (Putra, 2010).

Salah satu cara untuk memperoleh ciri pada citra dalam permasalahan pengenalan pola adalah dengan mengekstrak obyek pada citra dari *background*nya. Dengan memisahkan objek dengan *background* maka komputer akan lebih mudah dalam mengenali ciri pada citra

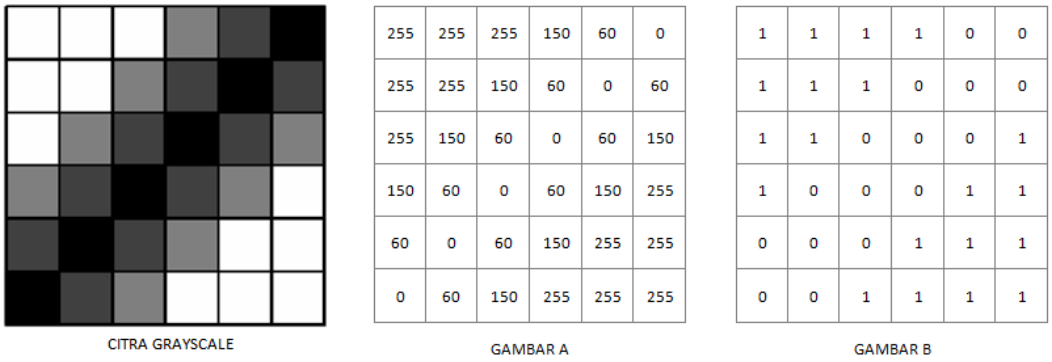
Menurut Prasetyo satu cara yang jelas untuk mengekstrak obyek dari *background* adalah dengan memilih *threshold* T yang membagi mode-mode ini. Sembarang titik (x, y) untuk dimana $f(x, y) \leq T$ disebut object point, sedangkan yang lain disebut background point.

$$g(x,y) = \begin{cases} 0,\ jika\ f(x,y) \leq T \\ 1,\ jika\ f(x,y) > T \end{cases} \tag{2.11}$$

Pixel yang diberi nilai 0 berkaitan dengan obyek sedangkan pixel yang diberi nilai 1 berkaitan dengan *background*.

Misalkan terdapat citra *grayscale* dengan ukuran 6 x 6 pixel, kemudian ubah citra menjadi rangkaian data numerik, sehingga dapat diolah dalam proses pengenalan huruf seperti pada Gambar 2.6. Gambar A mewakili matriks warna dengan nilai *gray level*-nya. Gambar B mewakili matriks yang merubah gambar A dengan nilai 0 untuk pixel yang *gray level*-nya lebih dari (*threshold* T) 128, dan nilai 1 untuk pixel yang *gray level*-nya kurang dari atau sama dengan (threshold T) 128.

Kemudian ubah matriks B menjadi sebuah vektor baris dengan nilai yang bersesuaian secara horisontal:



255	255	255	150	60	0
255	255	150	60	0	60
255	150	60	0	60	150
150	60	0	60	150	255
60	0	60	150	255	255
0	60	150	255	255	255

1	1	1	1	0	0
1	1	1	0	0	0
1	1	0	0	0	1
1	0	0	0	1	1
0	0	0	1	1	1
0	0	1	1	1	1

Gambar 2.6: Proses Ekstraksi Ciri

1	1	1	1	0	0	1	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	1	0	0	1	1	1	1

Gambar 2.7: Vektor Baris

Kemudian nilai array ini lah yang akan digunakan pada proses *backpropagation* untuk pengenalan pola.

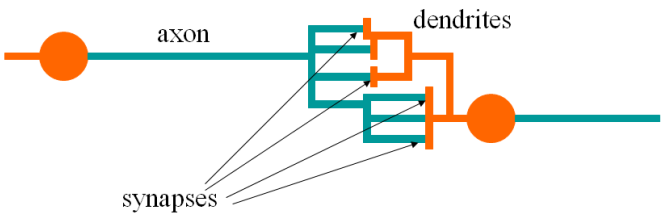
2.3 Jaringan Syaraf Tiruan *(Artificial Neural Network)*

2.3.1 Inspirasi Biologi

Jaringan syaraf tiruan merupakan bagian dari kecerdasan buatan yang terinspirasi dari jaringan syaraf otak manusia. Otak sama halnya seperti prosesor sederhana yang bertugas untuk memproses informasi.

Masing-masing sel tersebut berinteraksi mendukung kinerja otak. Setiap sel (neuron) memiliki satu nukleus (soma)yang bertugas untuk memproses informasi,kemudian informasi diterima oleh dendrit, dan disebarkan melalui akson. Pertemuan informasi antar syaraf berada di sinapsis.

Gambar 2.8: Syaraf Otak Manusia (sumber : www.fairuzelsaid.upy.ac.id)



Gambar 2.9: Ilustrasi Syaraf Manusia

Informasi yang dikirimkan oleh akson berupa rangsangan dengan sebuah batas ambang (*threshold*). Pada batas tertentu, syaraf lain akan teraktifasi dan merespon, hubungan antar syaraf terjadi secara dinamis. Otak manusia memiliki kemampuan untuk belajar dan beradaptasi sehingga mampu mengenali wajah, pola, mengkontrol organ tubuh baik secara disengaja maupun secara reflek berdasarkan pengalaman yang sebelumnya telah dipelajari oleh otak.

2.3.2 Definisi Jaringan Syaraf Tiruan

Jaringan Syaraf Tiruan (JST) didefinisikan sebagai suatu sistem pemrosesan informasi yang mempunyai karakteristik menyerupai jaringan syaraf manusia (Hermawan, 2006). Berikut ini adalah beberapa istilah dalam jaringan

syaraf tiruan yang sering ditemui

1. Neuron atau Node atau Unit: sel syaraf tiruan yang merupakan elemen pengolahan pada jaringan syaraf tiruan. Setiap neuron akan menerima data input, lalu memproses input tersebut kemudian mengirimkan hasilnya berupa sebuah output.

2. Jaringan: kumpulan neuron yang saling terhubung satu sama lain dan membentuk lapisan.

3. Lapisan tersembunyi (*hidden layer*): lapisan yang tidak berinteraksi dengan dunia luar. Lapisan ini mampu memperluas kemampuan jaringan syaraf tiruan dalam menghadapi maslah-masalah yang kompleks.

4. *Input*: sebuah nilai masukan yang kemudian akan diproses menjadi nilai keluaran.

5. *Output*: solusi atau hasil dari hasil proses nilai masukan.

6. Bobot: nilai matematis dari sebuah koneksi antar neuron.

7. Fungsi aktivasi: fungsi yang digunakan untuk mengupdate nilai-nilai bobot pada setiap iterasi dari semua nilai input.

8. Fungsi aktivasi sederhana adalah mengalikan input dengan bobotnya lalu kemudian menjumlahkannya (disebut penjumlahan sigma) berbentuk linier atau tidak linier, dan sigmoid.

9. Paradigma pembelajaran: bentuk pembelajaran.

Menurut Siang (2005:2), Jaringan Syaraf Tiruan (JST) adalah sistem pemroses informasi yang memiliki karakteristik mirip dengan jaringan syaraf biologi. JST dibentuk sebagai generalisasi model matematika dari jaringan syaraf biologi, dengan asumsi bahwa:

1. Terjadi pemrosesan informasi pada banyak elemen sederhana (neuron);
2. Sinyal dikirimkan diantara neuron melalui penghubung-penghubung;
3. Setiap penghubung antar neuron memiliki bobot yang dapat memperkuat atau memperlemah sinyal;
4. Untuk menentukan output, setiap neuron menggunakan fungsi aktivasi (biasanya bukan fungsi linier) yang dikenakan pada jumlahan input yang diterima. Besarnya output ini selanjutnya dibandingkan dengan suatu batas ambang.

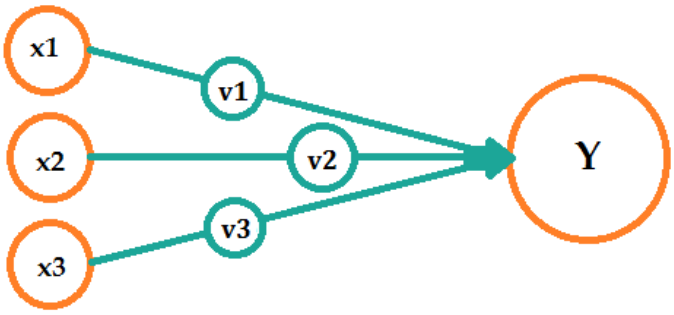
Jaringan syaraf tiruan ditentukan oleh 3 hal:

1. Pola hubungan antar neuron (disebut arsitektur jaringan).
2. Metode untuk menentukan bobot penghubung disebut metode training atau learning atau algoritma.
3. Fungsi aktivasi (fungsi transfer).

Neuron dalam jaringan syaraf tiruan sering diganti dengan istilah simpul. Setiap simpul tersebut berfungsi untuk mengirim atau menerima sinyal ke simpul-simpul lainnya. Pengiriman sinyal disampaikan melalui penghubung. Bobot merupakan kekuatan hubungan yang terjadi antara setiap simpul yang saling terhubung.

Model-model jaringan syaraf tiruan sangat ditentukan oleh arsitektur jaringan dan algoritma pelatihan. Arsitektur tersebut gunanya untuk menjelaskan arah perjalanan sinyal atau data di dalam jaringan. Sedangkan algoritma belajar menjelaskan bagaimana bobot koneksi harus diubah agar pasangan masukan-keluaran yang diinginkan dapat tercapai. Terdapat berbagai cara yang dapat dilakukan untuk merubah harga bobot koneksi, tergantung pada

jenis algoritma pelatihan yang digunakan. Dengan mengatur besarnya nilai bobot ini diharapkan dapat meningkatkan kinerja jaringan dalam mempelajari berbagai macam pola yang dinyatakan oleh setiap pasangan masukan-keluaran. Sebagai contoh, perhatikan neuron Y pada Gambar 2.10.



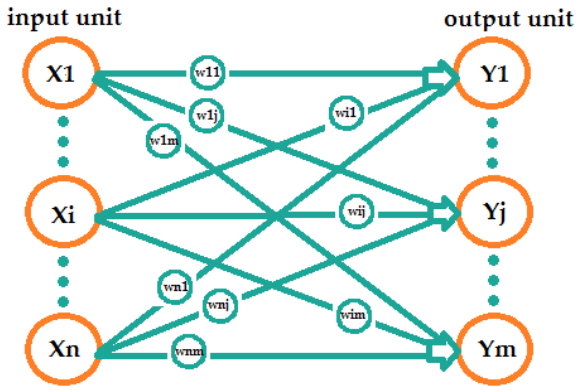
Gambar 2.10: Sebuah Sel Syaraf Tiruan

Y menerima *input* dari neuron $x1$, $x2$, dan $x3$ dengan bobot hubungan masing-masing adalah $v1$, $v2$, dan $v3$. Ketiga impuls neuron yang ada dijumlahkan $net = x1v1 + x2v2 + x3v3$. Besarnya impuls yang diterima oleh Y mengikuti fungsi aktivasi $y = f(net)$. Apabila nilai fungsi akivasi cukup kuat, maka sinyal akan diteruskan. Nilai fungsi aktivasi (keluaran model jaringan) juga dapat dipakai sebagai dasar untuk merubah bobot.

2.3.3 Arsitektur Jaringan

Jaringan Layar Tunggal

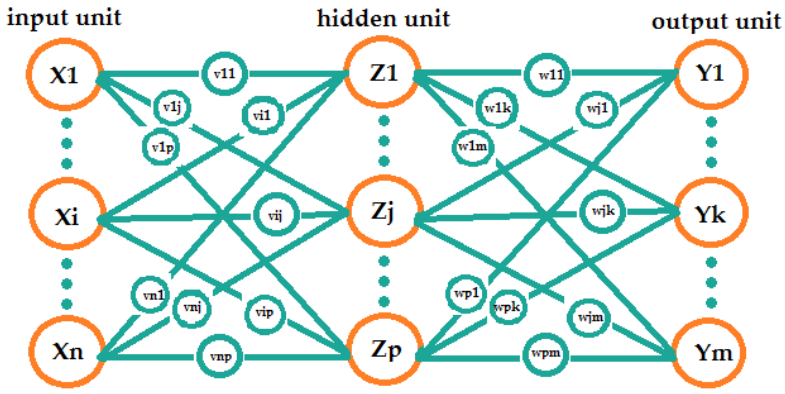
Jaringan layar tunggal merupakan sekumpulan *input neuron* dihubungkan langsung dengan sekumpulan *output*nya. Pada Gambar 2.11 diperlihatkan bahwa arsitektur jaringan layar tunggal dengan n buah masukan $(X_1, X_2, ..., X_n)$ dan m buah keluaran $(Y_1, Y_2, ..., Y_m)$. Dalam jaringan ini semua unit *input* dihubungkan secara langsung dengan semua unit *output*.



Gambar 2.11: Jaringan Layar Tunggal

Jaringan Layar Jamak

Jaringan layar jamak merupakan jaringan dengan n buah unit masukan $(X_1, X_2, ..., X_n)$, sebuah layar tersembunyi yang terdiri dari p buah unit $(Z_1, Z_2, ..., Z_p)$ dan m buah unit keluaran$(Y_1, Y_2, , Y_m)$. Dibandingkan dengan layar tunggal, jaringan layar jamak dapat menyelesaikan masalah yang lebih kompleks, meskipun kadangkala proses pelatihan lebih kompleks dan lama. Arsitektur jaringan layar jamak dapat dilihat pada Gambar 2.12.



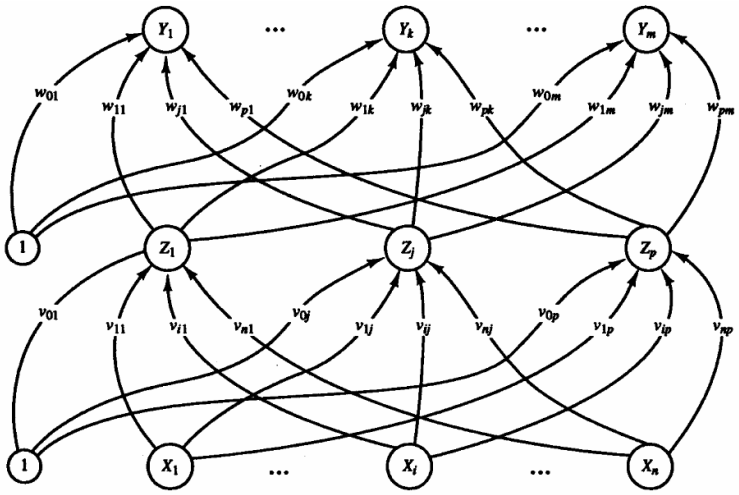
Gambar 2.12: Jaringan Layar Jamak

2.4 *Backpropagation*

Backpropagation merupakan salah satu metode dari jaringan syaraf tiruan yang dapat diaplikasikan dengan baik dalam bidang *forecasting* (peramalan). *Backpropagation* melatih jaringan untuk mendapatkan keseimbangan antara kemampuan jaringan mengenali pola yang digunakan selama training serta kemampuan jaringan untuk memberikan respon yang benar terhadap pola masukan yang serupa namun tidak sama dengan pola yang dipakai selama pelatihan (Siang, 2005:119).

Menurut Kusumadewi (2003:116), dalam pelatihan dengan *backpropagation* sama halnya seperti pelatihan pada jaringan syaraf yang lain. Pada jaringan umpan maju (*feedfoward*), pelatihan dilakukan untuk menghitung bobot sehingga pada akhir pelatihan akan diperoleh bobot-bobot yang baik. Selama proses pelatihan, bobot-bobot diatur secara iteratif untuk meminimumkan kesalahan (*error*) yang terjadi. Kesalahan dihitung berdasarkan rata-rata kuadrat kesalahan (MSE). MSE juga dijadikan dasar perhitungan untuk kerja fungsi aktivasi.

Sebagian besar pelatihan untuk jaringan *feedfoward* menggunakan gradient dari fungsi aktivasi untuk menentukan bagaimana mengatur bobot-bobot agar dapat meminimumkan kinerja. *Backpropagation* merupakan suatu teknik yang digunakan untuk menentukan *Gradient*. Pada dasarnya, algoritma pelatihan standar backpropagation akan menggerakkan bobot dengan arah gradient negatif. Prinsip dasar dari algoritma backpropagation yaitu memperbaiki nilai bobot-bobot jaringan. *Backpropagation* memiliki beberapa unit yang ada dalam satu atau lebih layar tersembunyi (Siang, 2005:98).



Gambar 2.13: Arsitektur *Backpropagation*

Gambar 2.13 adalah arsitektur *backpropagation* dengan n unit masukan (ditambah sebuah bias), sebuah layar tersembunyi yang terdiri dari p unit (ditambah sebuah bias), serta m unit keluaran.

V_{ij} merupakan bobot garis dari unit masukan X_i ke unit layar tersembunyi Z_j (V_{0j} merupakan bobot garis yang menghubungkan bias di unit masukan ke unit layar tersembunyi Z_j). W_{jk} merupakan bobot dari layar tersembunyi Z_j ke unit keluaran Y_k (W_{0k} merupakan bobot dari bias dilayar tersembunyi ke unit keluaran Z_k).

2.4.1 Fungsi Aktivasi

Setiap layer yang terdapat pada arsitektur jaringan saraf tiruan memiliki fungsi aktivasi. Fungsi aktivasi merupakan fungsi umum yang akan digunakan untuk membawa input pada layer input menuju layer diatasnya untuk mendapatkan output yang diinginkan. Fungsi aktivasi inilah yang akan merubah besarnya bobot.

Menurut J.J Siang (2005),fungsi aktivasi yang dapat digunakan dalam *back-*

propagation harus memenuhi beberapa syarat, yaitu : kontinu, terdiferensial dengan mudah dan merupakan fungsi yang tidak turun, agar fungsi tersebut dapat memperbaiki bobot-bobot jaringan sehingga meminimalkan kesalahan yang terjadi pada setiap iterasi dan pada akhir proses dapat diperoleh keluaran yang sesuai dengan target keluaran yang diinginkan.

Ada beberapa fungsi aktivasi yang biasa digunakan pada jaringan syaraf tiruan, yaitu :

1. Fungsi Sigmoid Biner

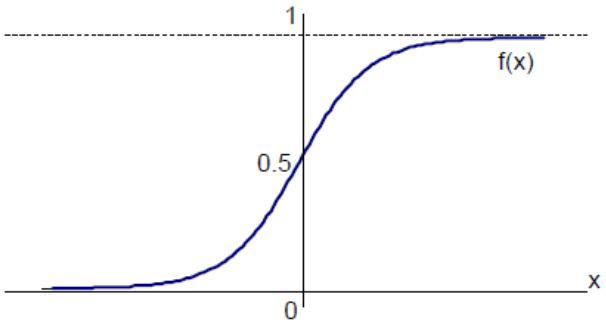
   Fungsi ini merupakan fungsi yang kontinu, terdiferensial dengan mudah, dan merupakan fungsi yang tidak turun, sehingga umum digunakan untuk jaringan syaraf tiruan yang menggunakan metode *backpropagation*. Fungsi sigmoid biner memiliki nilai antara 0 sampai 1.

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.12}$$

   Dengan turunan :

$$\begin{aligned} f(x) &= \frac{1}{1+e^{-x}} = (1+e^{-x})^{-1} \\ f'(x) &= -1.(1+e^{-x})^{-2}.-e^{-x} \\ &= \frac{e^{-x}}{(1+e^{-x})^2} \\ &= \left(\frac{1}{1+e^{-x}}\right)\left(\frac{e^{-x}}{1+e^{-x}}\right) \\ &= \left(\frac{1}{1+e^{-x}}\right)\left(\frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}}\right) \\ &= f(x)(1-f(x)) \end{aligned}$$

   Grafik fungsinya tampak pada gambar 2.14 :



Gambar 2.14: Fungsi Sigmoid Biner dengan Range $(0, 1)$

2. Fungsi Sigmoid Bipolar

   Fungsi ini hampir sama dengan fungsi sigmoid biner, hanya saja output dari fungsi ini memiliki range antara 1 sampai -1 seperti pada Gambar 2.15. Fungsi sigmoid bipolar dirumuskan sebagai :

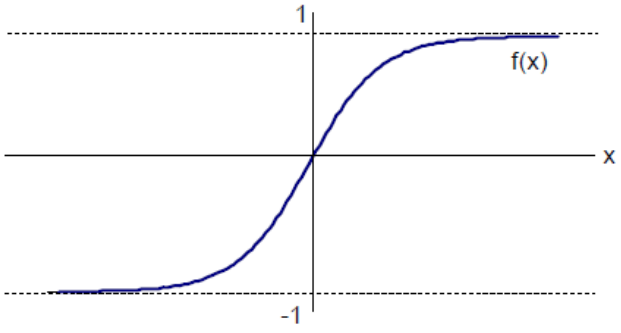
$$f(x) = \frac{2}{1 + e^{-x}} - 1 \tag{2.13}$$

   Dengan turunan :

$$\begin{aligned} f(x) &= \frac{2}{1 + e^{-x}} - 1 \\ &= \frac{2}{1 + e^{-x}} - \frac{1 + e^{-x}}{1 + e^{-x}} \\ &= \frac{2 - 1 - e^{-x}}{1 + e^{-x}} \\ &= \frac{1 - e^{-x}}{1 + e^{-x}} \\ f'(x) &= \frac{e^{-x}(1 + e^{-x}) + e^{-x}(1 - e^{-x})}{(1 + e^{-x})^2} \\ &= \frac{e^{-x} + e^{-2x} + e^{-x} - e^{-2x}}{(1 + e^{-x})^2} \end{aligned}$$

$$
\begin{aligned}
&= \frac{2e^{-2x}}{(1+e^{-x})^2} \\
&= \left(\frac{1}{1+e^{-x}}\right)\left(\frac{2+2e^{-x}-2}{1+e^{-x}}\right) \\
&= \frac{1}{2}\left(\frac{2}{1+e^{-x}}\right)\left(\frac{2(1+e^{-x})-2}{1+e^{-x}}\right) \\
&= \frac{1}{2}\left(\frac{2}{1+e^{-x}}\right)\left(\frac{2(1+e^{-x})}{1+e^{-x}}-\frac{2}{1+e^{-x}}\right) \\
&= \frac{1}{2}\left(1+\frac{2}{1+e^{-x}}-1\right)\left(1-\frac{2}{1+e^{-x}}+1\right) \\
&= \frac{1}{2}\left(1+\left(\frac{2}{1+e^{-x}}-1\right)\right)\left(1-\left(\frac{2}{1+e^{-x}}-1\right)\right) \\
&= \frac{1}{2}(1+f(x))(1-f(x))
\end{aligned}
$$

Grafik fungsinya tampak pada Gambar 2.15 :



Gambar 2.15: Fungsi Sigmoid Bipolar dengan Range $(-1, 1)$

Pada skripsi ini, fungsi aktivasi yang akan digunakan adalah fungsi simoid biner karena input jaringan merupakan data biner dan fungsi ini merupakan fungsi yang biasa digunakan untuk metode *backpropagation*.

2.4.2 Pelatihan Standar *Backpropagation*

Ada 3 fase pelatihan *backpropagation* menurut Siang (2005:100-104) yaitu.

Fase 1 (Propagasi Maju)

Setiap sinyal masukan propagasi maju dihitung maju dengan menggunakan fungsi aktivasi yang ditentukan dari layar input ke layar tersembunyi hingga layar keluaran. Selama tahap ini, setiap unit masukan (X_i) menerima sinyal masukan lalu mengirim sinyal ini ke setiap unit tersembunyi $Z_1..., Z_p$. Kemudian hitung aktivasinya untuk setiap unit tersembunyi lalu mengirim sinyalnya ke setiap unit keluaran. Setiap unit keluaran (Y_k) kemudian menghitung aktivasinya untuk menunjukkan respon jaringan terhadap pola masukan yang diberikan.

Fase 2 (Propagasi Mundur)

Kesalahan (selisih antara keluaran jaringan dengan target yang diinginkan) yang terjadi dipropagasi mundur mulai dari garis yang berhubungan langsung dengan unit-unit di layar keluaran. Untuk menentukan galat antara pola masukan dengan unit keluaran selama pelatihan, maka setiap unit keluaran membandingkan aktivasi Y_k dengan targetnya T_k . Setelah didapat galat, faktor $\delta_k(k = 1, ..., m)$ dihitung. δ_k digunakan untuk mendistribusikan galat pada unit keluaran Y_k kembali ke seluruh unit pada lapis sebelumnya (unit tersembunyi yang terhubung dengan Y_k).

Fase 3 (Perubahan Bobot)

Pada fase ini dilakukan modifikasi bobot guna menurunkan kesalahan yang terjadi. Galat yang diperoleh pada langkah 2 digunakan untuk mengubah bobot antara keluaran dengan lapisan tersembunyi. Menggunakan cara yang

sama, faktor $\delta_j (j = 1, ..., p)$ dihitung untuk setiap unit Z_j. Faktor δ_j digunakan untuk mengubah bobot seluruh antara lapisan tersembunyi dengan lapisan masukan. Bobot untuk seluruh lapisan langsung disesuaikan setelah faktor δ ditentukan. Penyesuaian bobot w_{jk} (dari unit tersembunyi Z_j ke unit keluaran Y_k) didasarkan pada faktor δ_k dan aktivasi dari unit Z_j. Penyesuaian bobot v_{ij} (dari unit masukan X_i ke unit tersembunyi Z_j) adalah didasarkan pada faktor δ_j dan aktivasi unit masukan X_i.

Ketiga fase tersebut diulang-ulang hingga kondisi penghentian dipenuhi. Umumnya kondisi penghentian yang sering dipakai adalah jumlah interasi atau kesalahan. Iterasi akan dihentikan jika jumlah iterasi yang dilakukan sudah melebihi jumlah maksimum iterasi yang ditetapkan, atau jika kesalahan yang terjadi sudah lebih kecil dari batas toleransi yang ditetapkan.

2.4.3 Algoritma Pelatihan

Menurut Siang (2005:102-104), algoritma pelatihan untuk jaringan dengan satu *hidden layer* (dengan fungsi aktivasi sigmoid biner) adalah sebagai berikut.

Langkah 0 : Inisialisasi semua bobot dengan bilangan acak kecil.

Langkah 1 : Jika kondisi penghentian belum terpenuhi, lakukan langkah 2-9.

Langkah 2 : Untuk setiap pasang data pelatihan, lakukan langkah 3-8.

Fase 1 : Propagasi Maju

Langkah 3 : Tiap unit masukan menerima sinyal dan meneruskannya ke unit tersembunyi diatasnya.

Langkah 4 : Hitung semua keluaran di unit tersembunyi $Z_j (j = 1, 2, ..., p)$:

$$Z_{netj} = v_{0j} + \sum_{i=1}^{n} X_i v_{ij} \tag{2.14}$$

$$Z_j = f(z_{netj}) = \frac{1}{1 + e^{-Z_{netj}}} \tag{2.15}$$

Langkah 5 : Hitung semua keluaran jaringan di unit $Y_k(k = 1, 2, ..., m)$:

$$Y_{netk} = w_{0k} + \sum_{j=1}^{p} Z_j w_{jk} \tag{2.16}$$

$$Y_k = f(Y_{netk}) = \frac{1}{1 + e^{-Y_{netk}}} \tag{2.17}$$

Fase 2 : Propagasi Mundur

Langkah 6 : Hitung faktor δ unit keluaran berdasarkan kesalahan di setiap unit keluaran $Y_k(k = 1, 2, ..., m)$:

$$\delta_k = (T_k - Y_k) f'(Y_{netk}) = (T_k - Y_k) Y_k (1 - Y_k) \tag{2.18}$$

δ merupakan unit kesalahan yang akan dipakai dalam perubahan bobot layar dibawahnya (langkah 7). Hitung suku perubahan bobot w_{jk} (yang akan dipakai nanti untuk merubah bobot w_{jk}) dengan laju percepatan α:

$$\delta w_{jk} = \alpha \delta_k z_j; k = 1, 2, ..., m; j = 0, 1, ..., p \tag{2.19}$$

Langkah 7 : Hitung faktor δ unit tersembunyi berdasarkan kesalahan disetiap unit tersembunyi $Z_j(j = 1, 2, , p)$:

$$\delta_{netj} = \sum_{k=1}^{m} \delta_k w_{jk} \tag{2.20}$$

faktor δ unit tersembunyi :

$$\delta_j = \delta_{netj} f'(Z_{netj}) = \delta_{netj} Z_j (1 - Z_j) \tag{2.21}$$

Hitung suku perubahan bobot v_{ij} (yang dipakai nanti untuk merubah bobot V_{ij}):

$$\delta v_{ij} = \alpha \delta_j X_i; j = 1, 2, ..., p; i = 0, 1, ..., n \tag{2.22}$$

Fase 3 : Perubahan Bobot

Langkah 8 : Perubahan bobot garis yang menuju ke unit keluaran:

$$w_{jk}(baru) = w_{jk}(lama) + \delta w_{jk} \tag{2.23}$$

Dari unit $ke-1$ sampai unit $ke-p$ dilapisan tersembunyi juga dilakukan *peng-update-an* pada bias dan bobotnya (i=1,2,,n; j=1,,p):

$$v_{ij}(baru) = v_{ij}(lama) + \delta v_{ij} \tag{2.24}$$

Langkah 9 : Test kondisi berhenti
Setelah pelatihan selesai dilakukan, jaringan dapat dipakai untuk pengenalan pola. Untuk menentukan keluaran jaringan, langkah propagasi maju (langkah 4 dan 5) saja yang dipakai.

Beberapa kasus pelatihan yang dilakukan memerlukan iterasi yang banyak sehingga membuat proses pelatihan menjadi lama. Parameter α atau laju pemahaman dapat digunakan untuk mempercepat proses iterasi. Nilai α terletak antara $0\,dan\,1(0 \leq \alpha \leq 1)$. Jika harga α semakin besar, maka iterasi yang dipakai semakin sedikit. Akan tetapi jika harga α terlalu besar, maka dapat merusak pola jaringan yang sudah benar sehingga pemahaman menjadi lambat.

Pemilihan bobot awal dapat mempengaruhi baik dan buruknya proses pelatihan, dalam standar *backpropagation*, bobot dan bias diisi dengan bilangan acak kecil dan biasanya bobot awal diinisialisasi secara random dengan nilai antara $-0,5$ sampai $0,5$ (atau -1 sampai 1 atau interval yang lainnya).

2.5 Simulasi Algoritma Pelatihan *Backpropagation*

Misalkan akan dibuat iterasi algoritma algoritma *Backpropagation* dengan sebuah *hidden layer* (3 unit) dengan 2 masukan X_1 dan X_2 ($X_1 = 1, X_2 = 1, T = 0$) serta laju pembelajaran $\alpha = 0.2$.

Penjelasan

Dari penjalasan diketahui terdapat 1 buah unit keluaran Y, 3 unit Z, 2 unit X serta 2 bias (untuk unit tersembunyi dan unit keluaran).

Mula-mula bobot diberi nilai acak yang kecil (range $[-1, 1]$). Misalkan bobot v_{ij} seperti tampak pada tabel berikut

Tabel 2.1: Tabel Bobot v_{ij}

	Z_1	Z_2	Z_3
X_1	0.2	0.3	-0.1
X_2	0.3	0.1	-0.1
1	-0.3	0.3	0.3

Tabel 2.2: Tabel Bobot w_{jk}

	Z_1	Z_2	Z_3	1
Y	0.5	-0.3	-0.4	-0.1

Langkah 4 : hitung keluaran unit tersembunyi (Z_j)

$Z_{netj} = v_{0j} + \sum_{i=1}^{n} X_i v_{ij}$

$Z_{net1} = -0.3 + 1(0.2) + 1(0.3) = 0.2$

$Z_{net2} = 0.3 + 1(0.3) + 1(0.1) = 0.7$

$Z_{net3} = -0.3 + 1(-0.1) + 1(-0.1) = 0.1$

$$Z_j = f(Z_{netj}) = \frac{1}{1 + e^{-Z_{netj}}}$$

$$Z_1 = \frac{1}{1 + e^{-0.2}} = 0.55 \; ; \; Z_2 = \frac{1}{1 + e^{-0.7}} = 0.67 \; ; \; Z_3 = \frac{1}{1 + e^{-0.1}} = 0.52$$

Langkah 5 : hitung keluaran unit Y_k

$Y_{netK} = w_{0j} + \sum_{j=1}^{p} Z_j w_{jk}$

Karena jaringan hanya memiliki sebuah unit keluaran Y, maka

$Y_{netk} = w_{01} + \sum_{k=1}^{m} Z_j w_{jk}$

$Y_{netk} = -0.1 + 0.559(0.5) + 0.67(-0.3) + 0.52(-0,4) = -0.24$

$$Y_k = f(Y_{netk}) = \frac{1}{1 + e^{-Y_{netk}}} = \frac{1}{1 + e^{-0.24}} = 0.44$$

Langkah 6 : Hitung faktor δ di unit keluaran Y_k

$\delta_k = (T_k - Y_k)f'(Y_{netk}) = (T_k - Y_k)Y_k(1 - Y_k)$

Karena jaringan hanya memiliki satu keluaran maka

$\delta_k = (T_k - Y_k)f'(Y_{netk}) = (T - Y)Y(1 - Y) = (0 - 0,44)(0,44)(1 - 0,44) = -0,11$

Suku perubahan bobot $w_{jk}(dengan \alpha = 0,2)$:

$\delta w_{jk} = \alpha \delta_k Z_j (k = 1; j = 0, 1, 2, 3)$

$\delta w_{01} = 0.2(-0.11)(1) = -0.02$

$\delta w_{11} = 0.2(-0.11)(0.55) = -0.01$

$\delta w_{21} = 0.2(-0.11)(0.67) = -0.01$

$\delta w_{31} = 0.2(-0.11)(0.52) = -0.01$

Langkah 7 : Hitung penjumlahan kesalahan dari unit tersembunyi (δ)

$\delta_{netj} = \sum_{k=1}^{m} \delta_k w_{jk}$.

Karena jaringan hanya memiliki sebuah unit keluaran maka

$\delta_{netj} = \delta w_{j1} (j = 1, 2, 3)$

$\delta_{net1} = (-0,11)(0,5) = -0,05$

$\delta_{net2} = (-0,11)(-0,3) = 0,03$

$\delta_{net3} = (-0,11)(-0,4) = 0,04$

Faktor kesalahan δ di unit tersembunyi :

$\delta_j = \delta_{netj} f'(Z_{netj}) = \delta_{netj} Z_j (1 Z_j)$

$\delta_1 = -0,05(0,55)(1 - 0,55) = -0,01$

$\delta_2 = 0,03(0,67)(1 - 0,67) = 0,01$

$\delta_3 = 0,04(0,52)(1 - 0,52) = 0,01$

Suku perubahan bobot ke unit tersembunyi

$\delta v_{ij} = \alpha \delta_j X_i \ (j = 1, 2, 3; i = 0, 1, 2)$

Tabel 2.3: Suku Perubahan Bobot ke Unit Tersembunyi

	Z_1	Z_2	Z_3
X_1	$\delta v_{11} = (0,2)(-0,01)(1) = -0,002 \approx 0$	$\delta v_{12} = (0,2)(0,01)(1) = 0,002 \approx 0$	$\delta v_{13} = (0,2)(0,01)(1) = 0,002 \approx 0$
X_2	$\delta v_{21} = (0,2)(-0,01)(1) = -0,002 \approx 0$	$\delta v_{22} = (0,2)(0,01)(1) = 0,002 \approx 0$	$\delta v_{23} = (0,2)(0,01)(1) = 0,002 \approx 0$
1	$\delta v_{01} = (0,2)(-0,01)(1) = -0,002 \approx 0$	$\delta v_{02} = (0,2)(0,01)(1) = 0,002 \approx 0$	$\delta v_{03} = (0,2)(0,01)(1) = 0,002 \approx 0$

Langkah 8 : Hitung semua perubahan bobot

Perubahan bobot unit keluaran:

$w_{jk}(baru) = w_{jk}(lama) + \delta w_{jk} \ (k = 1; j = 0, 1, 2, 3)$

Tabel 2.4: Perubahan Bobot Unit Keluaran

	Y
Z_1	$w_{11} = 0,5 - 0,01 = 0,49$
Z_2	$w_{21} = -0,3 - 0,01 = -0,31$
Z_3	$w_{31} = -0,4 - 0,01 = -0,41$
1	$w_{01} = -0,1 - 0,02 = -0,12$

Perubahan Bobot Unit Tersembunyi:

$v_{ij}(baru) = v_{ij}(lama) + \delta v_{ij} \ (i = 1, 2, 3; j = 0, 1, 2)$

Tabel 2.5: Perubahan bobot unit tersembunyi

	Z_1	Z_2	Z_3
X_1	$v_{11} = 0,2 + 0 = 0,2$	$v_{12} = 0,3 + 0 = 0,3$	$v_{13} = -0,1 + 0 = -0,1$
X_2	$v_{21} = 0,3 + 0 = 0,3$	$v_{22} = 0,1 + 0 = 0,1$	$v_{23} = -0,1 + 0 = -0,1$
1	$v_{01} = -0,3 + 0 = -0,3$	$v_{02} = 0,3 + 0 = 0,3$	$v_{03} = 0,3 + 0 = 0,3$

Iterasi terus dilakukan sampai batas pemberhentian tertentu. Kemudian bobot akhir pada pelatihan ini nantinya akan disimpan dan digunakan untuk membandingkan bobot citra latih dan citra uji, sehingga sistem dapat mengenali citra berdasarkan dengan proses perbandingan bobot.

BAB III

PEMBAHASAN

Pembuatan sistem pengenalan citra tulisan tangan melalui beberapa proses. Berikut adalah proses yang harus dilakukan.

3.1 Akuisisi Citra (*Image Acquitition*)

"Akuisisi citra (*Image Acquitition*) adalah proses pengambilan citra dengan menggunakan alat bantu pengambil gambar" (Siang, 2005:97). Gambar diambil dengan melakukan *scan* menggunakan *scanner merk HP Deskjet Ink Adventage 1515* pada kertas yang telah di tuliskan sebuah karakter tulisan tangan. Citra yang diperoleh berupa citra warna (RGB) seperti pada gambar berikut.

Gambar 3.1: Hasil Scan Tulisan Tangan

3.2 Studi Kasus

Studi Kasus yang digunakan adalah dengan mengambil citra tulisan tangan berupa angka 9 (sembilan) sampai 0 (nol) dari 10 orang yang berbeda. Tulisan tangan dibuat dengan menggunakan tangan kanan, yang dituliskan

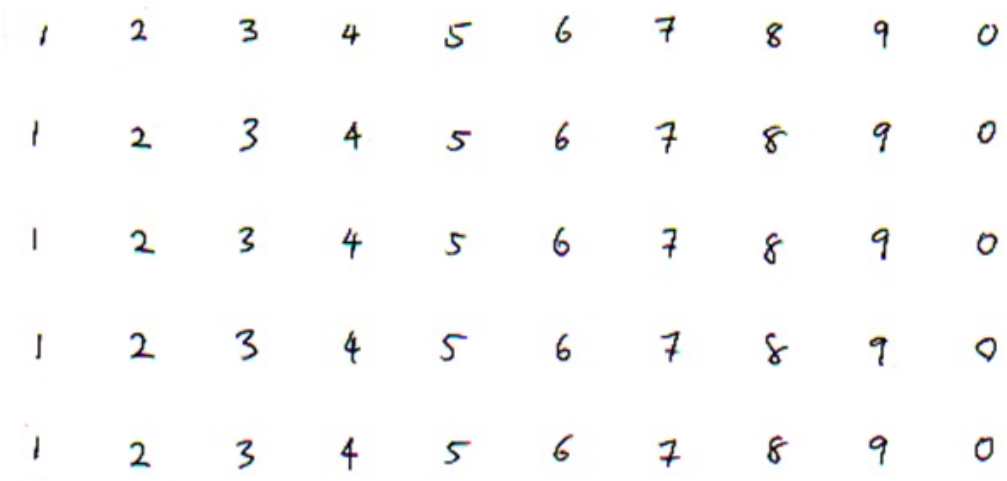
pada selembar kertas. Jumlah tulisan tangan yang diperoleh adalah sebanyak 100 buah.

3.3 Data Citra Tulisan Tangan

Data berupa tulisan tangan dituliskan pada kertas yang telah diperoleh kemudian di *scan* untuk memperoleh data digital, lalu data dikelompokan menjadi dua kelompok yang masing-masing terdiri dari 50 citra tulisan tangan digunakan sebagai citra latih, yang akan digunakan dalam proses pembelajaran jaringan. dan 50 citra tulisan tangan yang lain dimanfaatkan sebagai citra uji yang tidak melalui proses pelatihan.

3.3.1 Data Latih

Data citra latih akan digunakan untuk melatih sistem dalam mengenali pola citra untuk setiap karakter angka. Berikut 50 citra latih yang akan digunakan :



Gambar 3.2: Data Citra Latih

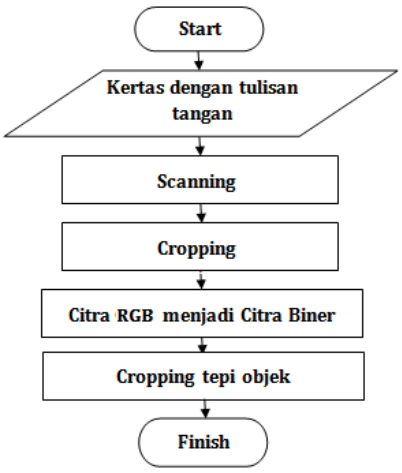
3.3.2 Data Uji

Setelah sistem mempelajari citra latih, kemudian data citra uji akan digunakan untuk menguji kemampuan sistem dalam pengenalan citra. Kemudian sistem akan membandingkan citra uji dengan citra latih yang telah dipelajari sebelumnya, untuk selanjutnya dapat melakukan pengenalan pola. Berikut 50 citra uji yang akan digunakan :

Gambar 3.3: Data Citra Uji

3.4 Pengolahan Data Citra

Pengolahan citra ini dilakukan setelah diperoleh data citra yang merupakan hasil dari *Scanning* pada kertas bertuliskan angka. Serangkaian proses pengolahan citra yang dilakukan pada penilitian ini meliputi proses *cropping*, mengubah citra RGB menjadi citra biner, cropping tepi objek. *Flowchart pre processing* dapat dilihat pada Gambar 3.4.



Gambar 3.4: *Flowchart* Pengolahan Citra

3.4.1 *Cropping*

Cropping adalah proses memotong citra yang bertujuan untuk mengambil bagian yang dibutuhkan dari citra. Gambar tulisan tangan yang diambil merupakan gambar yang umum, itu berarti tidak semua komponen pada citra dibutuhkan pada sistem yang akan dirancang, sehingga perlu dilakukan proses *Cropping*.

3.4.2 Mengubah Citra RGB Menjadi Citra Biner

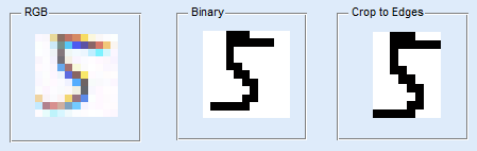
pada citra biner ini piksel dengan derajat keabuan yang lebih kecil dari nilai batas yang ditentukan akan diberikan nilai 0 (mewakili objek), sebaliknyan untuk piksel yang memiliki derajat keabuan lebih besar dari batas maka akan diberi nilai 1 (*mewakili background*).

3.4.3 *Cropping* Tepi Objek

memotong tepi pada citra bertujuan untuk memisahkan citra dengan latar belakangnya. proses ini akan mempermudah proses ekstraksi citra serta

pengenalan citra.

Berikut adalah hasil dari proses pengolahan citra :



Gambar 3.5: Hasil Pengolahan Citra

3.4.4 *Feature Extraction* (Ekstraksi Ciri)

Pada proses ini, citra masukan dikodekan menurut pixel-nya. Tujuan dari proses ini yaitu untuk memberi kode yang berbeda pada setiap objek, sehingga setiap objek dapat dipisahkan berdasarkan kode atau karakteristik yang dimiliki. Selain itu, proses pengkodean karakter ini juga bertujuan untuk mengambil ciri tertentu dari setiap setiap objek. Citra biner akan membentuk matriks pixel, komponen matrik yang bernilai 0 mewakili objek sedangkan nilai 1 mewakili *background* pada citra. berikut adalah matriks pixel yang dihasilkan pada matlab.

```
1   1   1   0   0   0   1   1   1   1   1
1   1   1   0   0   0   0   0   0   1   1
1   1   1   0   1   1   1   1   1   1   1
1   1   1   0   1   1   1   1   1   1   1
1   1   1   0   0   1   1   1   1   1   1
1   1   1   1   0   0   1   1   1   1   1
1   1   1   1   1   0   0   1   1   1   1
1   1   1   1   1   1   0   1   1   1   1
1   1   1   1   1   0   0   1   1   1   1
1   0   0   0   0   0   1   1   1   1   1
1   1   1   1   1   1   1   1   1   1   1
```

Gambar 3.6: Matriks Pixel yang Dihasilkan Matlab

3.5 Perancangan Sistem

Pada tahap ini akan dipersiapkan segala sesuatu yang dibutuhkan pada saat pembuatan sistem. Tahapan perancangan sistem terdiri dari beberapa proses sebagai berikut.

3.5.1 Perancangan Sistem Pelatihan

Sistem pelatihan merupakan proses awal dari pembuatan sistem pengenalan citra tulisan tangan. Pada proses ini akan disiapkan berbagai informasi yang akan digunakan dalam proses pengenalan. Berikut adalah beberapa tahapan penting dalam perancangan sistem pelatihan.

Arsitektur Jaringan Syaraf Tiruan

Pada skripsi ini penulis akan melakukan percobaan untuk beberapa variasi arsitektur. Variasi dilakukan terhadap jumlah neuron *hidden layer* dan *learning rate*. Sedangkan jumlah epoch maksimal yang ditentukan adalah 1000. Berikut adalah beberapa variasi yang akan dilakukan pada proses pembuatan jaringan.

Tabel 3.1: Variasi Arsitektur Jaringan

α	Jumlah Neoron *Hidden Layer*						
0.05	28	30	32	34	36	38	40
0.1	28	30	32	34	36	38	40

Pada setiap variasi arsitektur jaringan yang telah ditentukan, akan dilakukan percobaan pembuatan jaringan untuk mendapatkan rata-rata kuadrat kesalahan (MSE). Variasi Arsitektur dengan nilai MSE terkecil akan digunakan untuk pembuatan sistem pengenalan citra tulisan tangan.

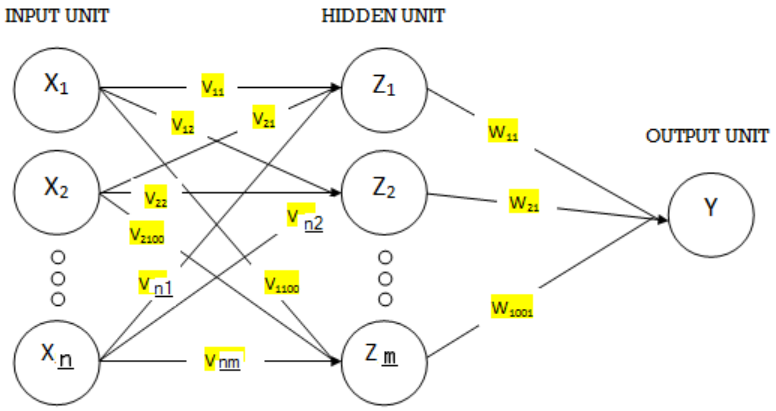
Supervised learning merupakan paradigma pembelajaran yang dirancang

pada JST, karena pada sistem ini terdapat pasangan data yaitu data masukan masukan dan target keluaran yang dipakai untuk melatih jaringan.

Pada pembobotan awal, dipilih bobot secara random, yaitu dengan membangkitkan bilangan-bilangan acak yang terletak pada range antara 0 sampai 1. Arsitektur jaringan yang digunakan pada skripsi ini adalah Arsitektur Jaringan Layar Jamak dengan 1 buah *Hidden Layer*.

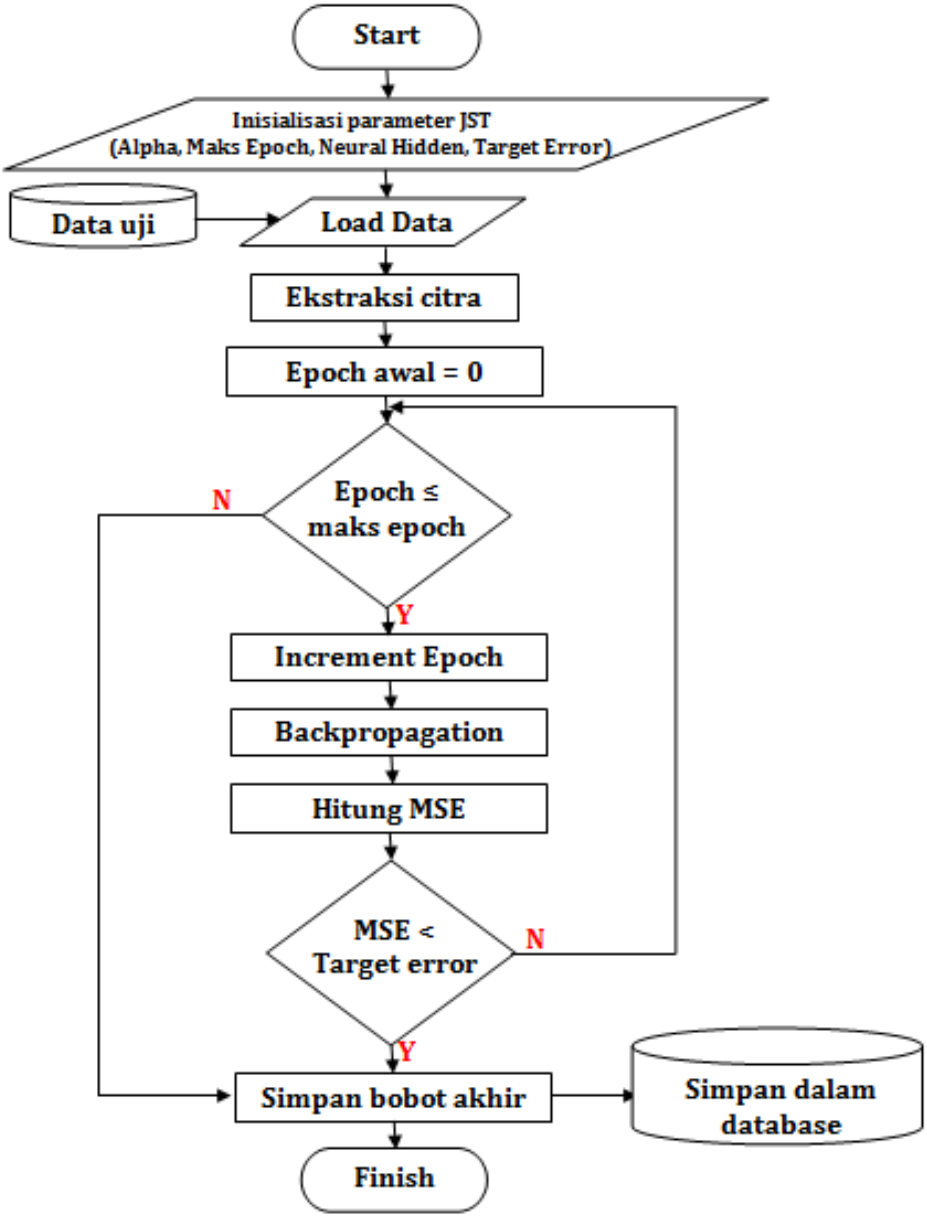
Arsitektur Jaringan *Backpropagation*

Arsitektur jaringan *Backpropagation* layar jamak dengan 1 hidden layer yang dirancang dalam penelitian ini dapat dilihat pada Gambar 3.7. X adalah nilai *input* merupakan nilai dari setiap pixel citra, terdiri dari n buah masukan (tergantung pada ukuran citra), Z merupakan hidden layer yang terdiri dari m buah (tergantung pada variasi saat pembuatan jaringan), serta Y yang merupaka *output* terdiri dari 1 target keleuaran



Gambar 3.7: Arsitektur Jaringan *Backpropagation*

Flowchart untuk langkah pelatihan ini dapat dilihat pada Gambar 3.8.



Gambar 3.8: *Flowchart* Proses Pelatihan *Backpropagation*

3.5.2 Perancangan Sistem Pengenalan

Pada proses ini perlu disiapkan sebuah komputer dengan spesifikasi tertentu, agar dapat dibuat sistem pengenalan citra tulisan tangan, berikut adalah spesifikasi komputer yang digunakan untuk pembuatan sistem.

Operating System	:	Windows 7 Ultimate 32-bit
Processor	:	Intel Core i5
Memory	:	2048 MB
Monitor	:	14 inch
Mouse dan Keyboard		

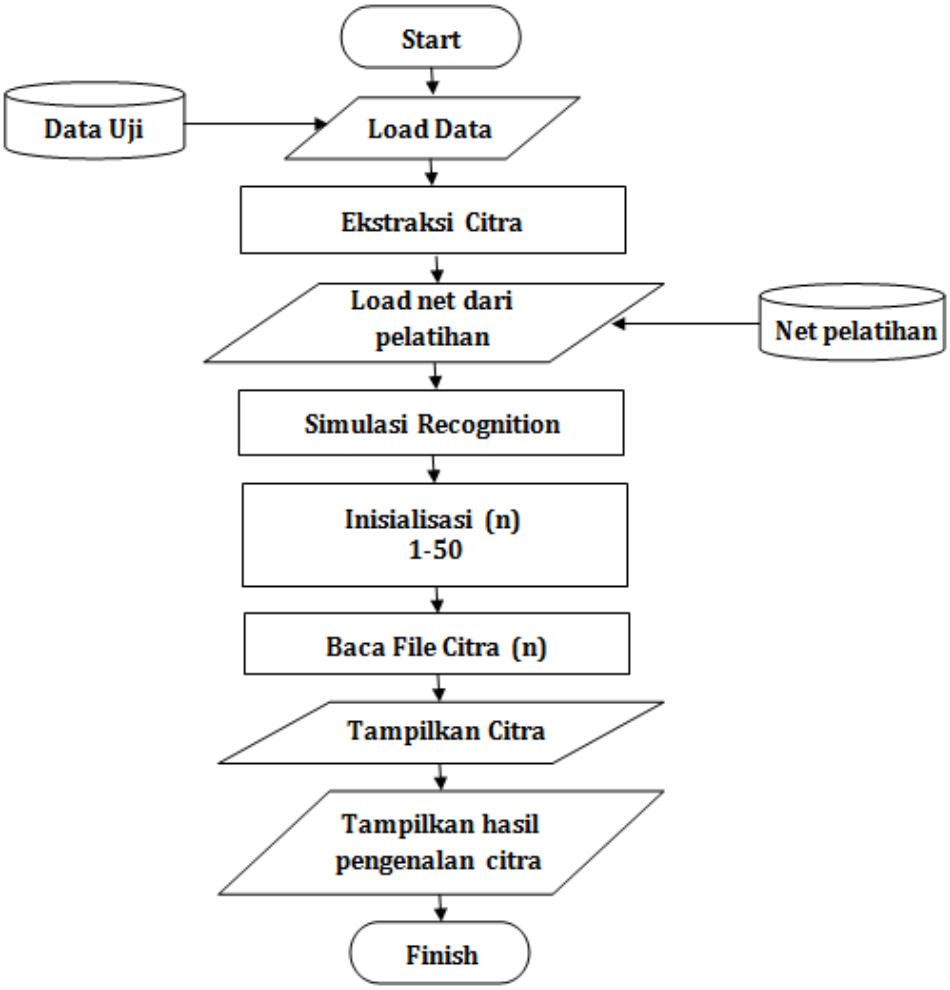
Proses pengenalan oleh sistem memiliki tahapan yang hampir sama dengan proses pembelajaran, yaitu dimulai dengan pengolahan citra uji, lalu mengambil ciri dari citra uji melalui proses *featuring extraction*, lalu kemudian dilakukan perhitungan bobot untuk kemudian dibandingkan dengan bobot yang diperoleh dari citra latih.

Perancangan Sistem Pengenalan Backpropagation

Apabila pelatihan jaringan backpropagation telah menghasilkan nilai keluaran yang sesuai dengan yang ditargetkan, maka sistem sudah dapat digunakan untuk proses pengenalan data input tulisan tangan yang sudah dilatih (data latih) maupun data baru yang belum dilatih atau data uji. Data uji ini merupakan citra tulisan tangan baru yang sebelumnya belum pernah dipelajari oleh sistem.

Pada proses pengenalan citra dengan metode *backprogation* akan digunakan nilai bobot keluaran dari proses pelatihan menggunakan citra latih, kemudian selanjutnya bobot tersebut dicocokkan dengan nilai bobot keluaran dari citra

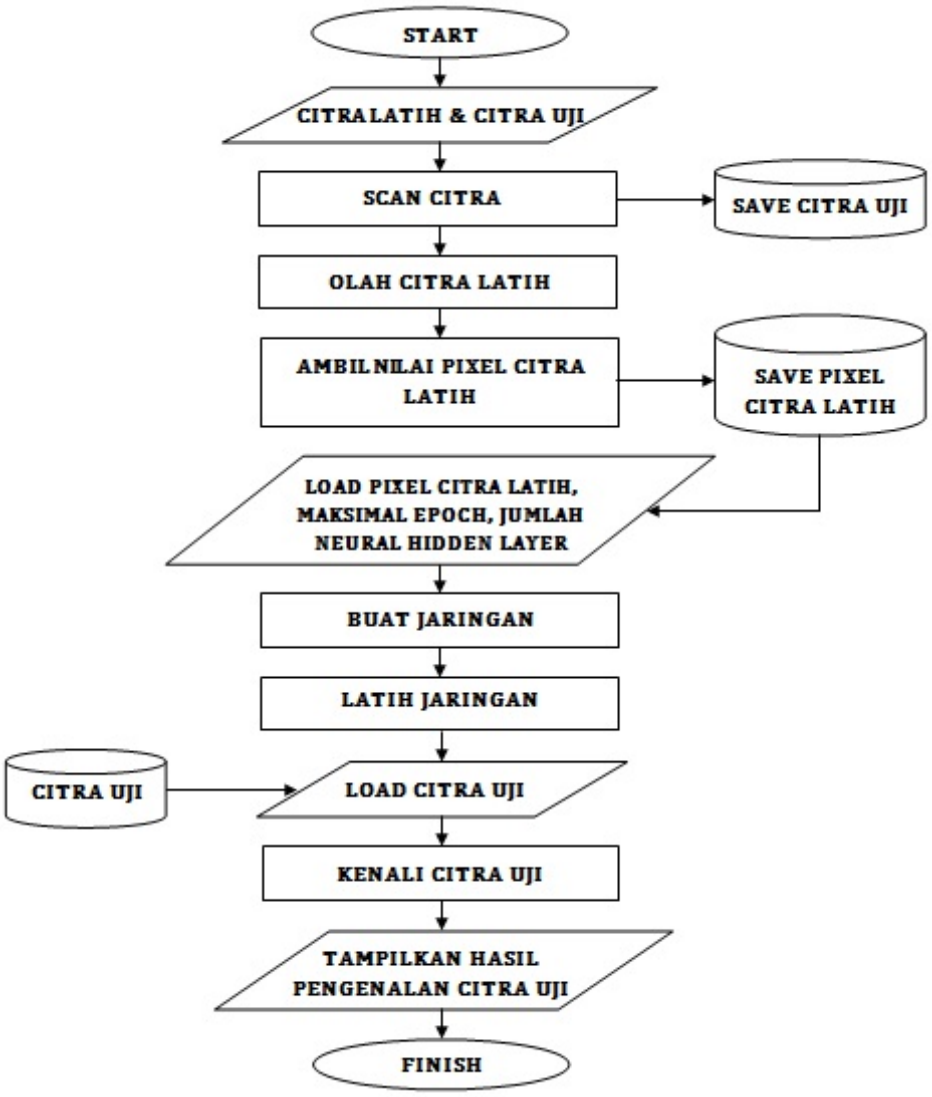
baru (data uji). Jadi data citra tulisan tangan baru yang diujikan juga akan mengalami proses pencarian nilai bobot keluaran. Berikut adalah *flowchart* untuk sistem pengenalan citra backpropagation.



Gambar 3.9: *Flowchart* Sistem Pengenalan Citra *Backpropagation*

3.6 Algoritma Pembuatan Sistem Pada Matlab

Algoritma yang ditampilkan adalah berupa diagram alir yang merupakan alur kerja proses pembuatan sistem pengenalan citra. Dimulai dari proses scanning sampai dengan menampilkan hasil pengenalan citra.



Gambar 3.10: Algoritma Pembuatan Sistem Pengenalan Citra

3.7 Proses Pembuatan Sistem

Aplikasi pengenalan citra tulisan tangan ini akan dibuat menggunakan *software* matlab. Tahapannya meliputi pengambilan nilai pixel dari setiap citra latih yang kemudian digunakan untuk pembuatan jaringann, lalu pembuatan form pengenalan dengan menggunakan GUI pada matlab, kemudian menggunakan jaringan yang telah dibuat sebelumnya untuk di uji kemampuannya dalam pengenalan citra tulisan tangan dengan menggunakan data citra uji.

3.7.1 Pengambilan Nilai Pixel dari Citra Data Latih

Citra data latih yang digunakan sebanyak 50 tulisan tangan, yang terdiri dari 10 karakter angka yang dituliskan oleh 5 orang berbeda. Ukuran matriks akan tergantung pada ukuran citra yang digunakan. Berikut adalah salah satu matriks pixel yang terbentuk.

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Matriks diatas memiliki ukuran 11 x 11, matriks diubah menjadi vektor baris berukuran 1x121. Merubah dimensi matriks 11x11 ini diperlukan karena data

input yang digunakan oleh matlab harus berbentuk vektor baris. Kemudian vektor baris akan digunakan untuk pembuatan jaringan.

[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 0 1 1 1 1 0 1 0
0 1 1 0 0 1 1 1 0 1 0 0 1 1 1 0 0 1 0 0 1 1 0
1 1 1 1 0 0 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1]

Matriks pixel disimpan dalam Microsoft Excel, lalu kemudian di *import* ke matlab untuk digunakan sebagai *input* pada proses pembuatan dan pelatihan jaringan.

3.7.2 Pembuatan Jaringan

Setelah di peroleh matriks pixel dari citra latih, kemudian akan dilakukan pembuatan jaringan pelatihan. Pada sub bab sebelumnya penulis telah melakukan perancangan variasi *learning rate* dan jumlah neuron *hidden layer* pada pembuatan jaringan. Berikut adalah MSE yang dihasilkan, serta waktu pelatihan yang dibutuhkan untuk setiap variasi.

Tabel 3.2: Nilai MSE yang Diperoleh dari Variasi Jaringan

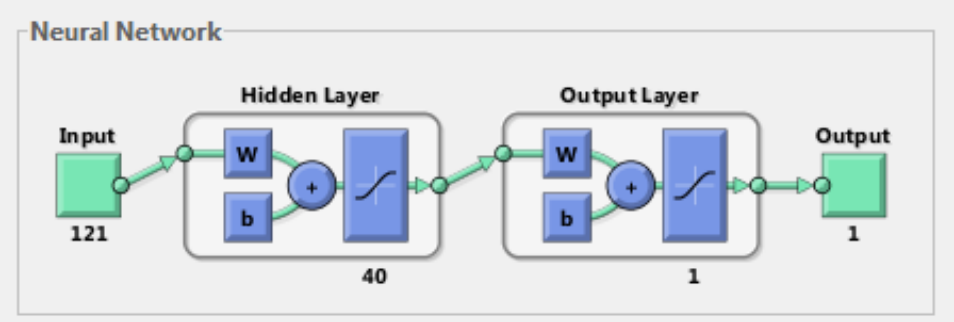
neuron hidden layer	Waktu Pelatihan (detik)	Epoch		MSE	
		$\alpha = 0.05$	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.1$
28	51	25	23	$5.76e-14$	$1.04e-09$
30	70	22	20	$5.31e-14$	$4.07e-10$
32	75	23	22	$4.99e-14$	$7.92e-10$
34	82	23	20	$5.95e-14$	$2.58e-09$
36	98	20	20	$5.39e-14$	$1.84e-10$
38	101	21	20	$5.88e-14$	$6.40e-10$
40	123	22	22	$4.48e-14$	$8.65e-10$

Dari hasil percobaan variasi, didapatkan bahwa jumlah neoron *hidden layer* dan *learning rate*, dapat mempengaruhi nilai MSE yang didapat. Jaringan semakin baik apabila nilai MSE semakin rendah. Namun semakin banyak Jumlah neoron *hidden layer* maka waktu yang dibutuhkan untuk melatih jaringan akan semakin lama.

Berdasarkah hasil percobaan yang dilakukan, penulis menggunakan arsitektur yang menghasilkan MSE terendah. Berikut adalah arsitektur jaringan yang digunakan.

jumlah neouron *hidden layer*	:	40
maksimal epoch	:	1000
learning rate	:	0.05
data input	:	matriks pixel citra tulisan tangan
output target	:	1,2,3,4,5,6,7,8,9,10

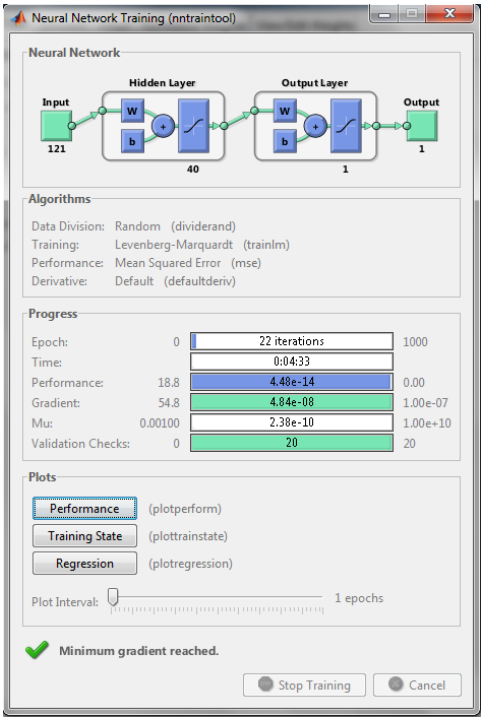
Berikut adalah tampilan arsitektur jaringan yang dibuat pada matlab :



Gambar 3.11: Arsitektur Jaringan pada Matlab

Jaringan yang telah dibuat kemudian dilatih. Dalam proses pelatihan, jaringan secara otomatis akan melakukan iterasi dan perhitungan MSE. Proses pelatihan jaringan akan berhenti apabila iterasi telah mencapai batas maksimal iterasi, atau MSE telah mencapai batas minimal MSE, atau eror telah mencapai target eror. Setelah dilatih, jaringan di export dengan nama "data" lalu kemudian digunakan untuk melakukan pengenalan citra tulisan.

berikut adalah tampilan untuk proses pelatihan jaringan pada matlab.



Gambar 3.12: Proses Pelatihan Jaringan

3.7.3 Pembuatan *Graphical User Interface* (GUI)

Setelah pembuatan jaringan, langkah berikutnya adalah pembuatan sistem. Apikasi pengenalan citra dibuat menggunakan *Graphical User Interface* (GUI) pada matlab.proses pengolahan citra yang terdapat pada sistem adalah meliputi proses *Pre pocessing* (*cropping, grayscale*, perbaikan kualitas), serta *featuring Extraction* atau ekstraksi ciri. Hasil dari proses ekstrasi citra ini nantinya akan digunakan oleh sistem sebagai input untuk melakukan pengenalan tulisan tangan. Waktu yang dibutuhkan sistem untuk melakukan pengenalan rata-rata 3 detik. Proses pengenalan citra menggunaka njaringan yang sebelumnya telah dibuat dan dilatih. Berikut adalah tampilan sistem yang telah dibuat.

Gambar 3.13: Rancangan Tampilan Sistem

Keterangan Tombol		
Load Image	:	Menginput citra dari komputer yang kemudian akan ditampilkan pada kolom dibawahnya
Select	:	Melakukan seleksi pada citra yang telah di input
Crop	:	Memotong object yang telah di select
Pre Processing dan features extraction	:	Melakukan *Per Processing* serta pengambilan pixel pada citra
recognize	:	Proses pengenalan citra

3.8 Proses Pengenalan Citra Tulisan Tangan oleh Sistem

Proses pada citra uji sama dengan yang dilakukan pada citra latih, yaitu pengolahan citra (*cropping*, RGB ke biner, *cropping* tepi citra), lalu lakukan ekstraksi citra untuk mendapatkan nilai pixelnya. Setelah itu dilakukan perhitungan oleh sistem menggunakan algoritma *backpropagation*. Nilai bobot

akhir yang didapat kemudian dibandingkan dengan semua bobot citra latih yang telah dipelajari sistem. Kemudian sistem akan mengenali citra berdasarkan kemiripan bobot dan pola citra. Berikut adalah tampilan saat sistem dijalankan

Gambar 3.14: Tampilan Saat Sistem Djalankan

Untuk penjabaran hasil pengenalan oleh sistem, akan dijelaskan pada bab selanjutnya.

BAB IV

HASIL

Pada bab ini akan ditunjukan hasil pengujian sistem dalam mengenali citra tulisan tangan setelah dilakukan input berupa 50 citra latih dan 50 citra uji pada sistem. Hasil yang didapat berupa persentase keberhasilan sistem dalam mengenali tulisan tangan. Kemudian akan dilakukan pula analisis terhadap hasil pengenalan oleh sistem.

4.1 Hasil Arsitektur Jaringan Syaraf Tiruan Backpropagation

Setelah sistem dijalankan maka akan terlihat apakah sistem mampu mengenali citra tlisan tangan yang telah di-*input*. Dengan mengamati hasil pengenalan tersebut, maka dapat diketahui berapa banyak citra yang mampu dikenali oleh sistem. Arsitektur jaringan yang digunakan pada sistem adalah α (*Learning Rate*) sebesar 0.05 serta jumlah *Neuron hidden layer* adalah sebanyak 40 unit neuron. Jumlah citra *input* adalah 50 citra latih dan 50 citra uji. Dari jumlah citra tulisan tangan yang mampu dikenali oleh sistem, dapat dihitung *recognition rate* nya dengan rumus

$$recognition\ rate = \frac{jumlah\ citra\ yang\ berhasil\ dikenali}{jumlah\ citra\ yang\ di\ uji} * 100\%$$

Analisis hasil dapat dilihat pada Tabel berikut :

Tabel 4.1: Analisis Hasil Variasi Arsitektur JST Backpropagation

α	Jumlah Neuron Hidden Layer	Epoch	MSE	Waktu Pelatihan Jaringan	Citra Latih yang Dikenali	Citra Uji yang Dikenali	Recognition Rate
0.05	40	22	$4.48e-14$	123 detik	50	46	96%

4.2 Hasil Pengenalan Citra Oleh Sistem

Citra yang di-*input* dalam sistem untuk dikenali merupakan citra yang sebelumnya telah di-*training* (citra latih) dan citra yang belum di-*training* (citra uji). Output dari sistem akan memperlihatkan bahwa sistem lebih mampu mengenali citra latih, karena sebelumnya sistem telah mengenali dan mempelajari pola citra. Berikut hasil dari pengenalan oleh sistem.

Tabel 4.2: Hasil Pengenalan Citra Latih

	Citra Angka									
orang ke -	1	2	3	4	5	6	7	8	9	0
1	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
3	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
4	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
5	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Tabel 4.3: Hasil Pengenalan Citra Uji

	Citra Angka									
orang ke -	1	2	3	4	5	6	7	8	9	0
1	✓	✓	✓	✓	x	✓	✓	✓	✓	✓
2	✓	✓	✓	x	✓	✓	✓	✓	✓	✓
3	✓	✓	✓	✓	x	✓	✓	✓	✓	✓
4	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
5	✓	✓	✓	x	✓	✓	✓	✓	✓	✓

Dari hasil pengenalan yang telah di lakukan oleh sistem dan telah dijelaskan pada tabel diatas, maka dapat dihitung total citra yang berhasil dan gagal dikenali sistem adalah sebagai berikut.

Tabel 4.4: Total Hasil Pengenalan

	Berhasil Dikenali	Gagal Dikenali
Citra Latih	50	0
Citra Uji	46	4
TOTAL	96	4

Berikutnya akan dihitung persentase keberhasilan sistem dalam pengenalan citra tulisan tangan per angka dengan rumus :

$$recognition\ rate = \frac{jumlah\ citra\ yang\ berhasil\ dikenali}{jumlah\ citra\ yang\ di\ uji} * 100\%$$

Tabel 4.5: Persentase Keberhasilan Sistem Mengenali Angka

Citra Angka	Jumlah Citra	Berhasil Dikenali	*Recognition Rate*
1	10	10	100%
2	10	10	100%
3	10	10	100%
4	10	8	80%
5	10	8	80%
6	10	10	100%
7	10	10	100%
8	10	10	100%
9	10	10	100%
0	10	10	100%

Dari tabel diatas dapat disimpulkan bahwa angka yang paling mudah untuk dikenali sistem adalah angka 1, 2, 3, 6, 7, 8, 9, 0 dengan *Recognition Rate* sebesar 100%. dan angka yang paling sulit dikenali sistem adalah angka 4 dan 5 dengan *Recognition Rate* sebesar 80%.

4.3 Analisis Hasil Kerja Sistem

Dari hasil pengujian sistem diperoleh *recognition rate* pengenalan tulisan tangan metode backpropagation sebesar 96%. Akan dijelaskan penyebab sistem mengalami kesalahan dalam mengenali citra, sistem dianggap gagal apa-

bila mengenali citra tulisan tangan tidak sesuai dengan tulisan tangan yang diujikan.

Berikut adalah beberapa penyebab kegagalan sistem dalam mengenali citra tulisan tangan.

1. Ketajaman dan Ketebalan Citra

   Tulisan tangan yang terlalu tajam dan tebal akan menyebabkan tingkat kontras pada citra lebih tinggi. Tulisan tangan yang tidak terlalu tajam dan tebal akan menyebabkan tingkat kontras pada citra lebih rendah, sehingga akan mempengaruhi saat dilakukan ekstraksi ciri.

2. Ukuran dan Bentuk Tulisan Tangan

   Pada saat pangambilan data tulisan tangan, biasanya setiap orang memiliki ukuran dan bentuk tulisan yang tidak sama sehingga menghasilkan data citra yang tidak homogen. Ukuran citra yang tidak seragam, dapat menyebabkan hasil pengenalan yang kurang sesuai.

3. Persentase Keutuhan Setiap Digit Citra

   Pada sistem terdapat menu untuk melakukan seleksi. Menu ini dugunakan untuk memilih tulisan tangan mana yang ingin dikenali. Pada seleksi ini, citra tulisan tangan harus terseleksi dengan sempurna (tidak ada bagian objek tulisan yang terpotong) dari setiap digit agar tetap dapat dikenali dengan benar. Apabila bentuk angka yang terseleksi tidak sempurna, kemungkinan besar sistem akan salah dalam mengenali. Hal tersebut terjadi karena nilai pixel yang didapat akan berubah dan berbeda dengan pola citra latih.

4. Bobot

   Pada sistem, jaringan akan melakukan perhitungan bobot untuk setiap input citra. Bobot yang diperoleh kemudian akan dibandingkan dengan

bobot dari setiap citra latih, sistem akan mengenali citra berdasarkan dengan bobot tersebut. Kesalahan sistem dalam mengenali tulisan tangan dimungkinkan karena bobot dari 1 digit (misal angka 5) lebih mirip dengan bobot citra lain (selain 5).

Kondisi-kondisi ini akan mempengaruhi tingkat keberhasilan sistem dalam mengenali citra tulisan tangan yang dilatihkan ataupun yang diujikan, sehingga dapat menyebabkan *recognition rate* menjadi lebih rendah dan terjadi kesalahan pada proses pengenalan.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari proses pengolahan citra, perancangan, pembuatan, sampai dengan pengujian sistem menggunakan metode *backpropagation* pada penelitian ini, didapatkan simpulan sebagai berikut.

1. Proses pengenalan citra tulisan tangan dimulai dengan mengambil nilai pixel dari citra uji dan citra latih yang merupakan input sistem. Kemudian dilakukan pembuatan jaringan serta pembuatan tampulan sistem dengan menggunakan *Graphical User Interface* (GUI) pada matlab. Setelah jaringan dilatih, maka jaringan dapat digunakan oleh sistem untuk mengenali citra tulisan tangan.

2. nilai parameter yang digunakan dalam proses pembelajarannya sangat mempengaruhi kemampuan sistem dalam mengenali citra tulisan tangan. Pada penelitian ini parameter terbaik yang digunakan learning rate = 0,05 dan jumlah hidden layer = 40 serta 1000 maksimal epoch.

3. Didapat 96 citra berhasil dikenali oleh sistem dengan *recognition rate* sebesar 96% dari total pengenalan citra sebanyak 50 citra latih dan 50 citra uji. Diperoleh waktu training selama 123 detik dan waktu pengenalan selama 2 detik.

4. Dari hasil pengujian, diperoleh angka yang paling mudah dikenali oleh sistem adalah angka 1, 2, 3, 6, 7, 8, 9, 0 dengan *Recognition Rate* sebesar

100%, dan angka yang paling sulit dikenali sistem adalah angka 4 dan 5 dengan *Recognition Rate* sebesar 80%.

5.2 Saran

Dari proses pengolahan citra, perancangan, pembuatan, sampai dengan pengujian sistem menggunakan metode backpropagation, saran yang dapat ditulis untuk penelitian ini adalah sebagai berikut.

- Dapat digunakan variasi-variasi arsitektur jaringan lain dalam pembuatan jaringan, dengan mengatur jumlah neuron *hidden layer* maupun *learning rate* .
- Pada penelitian berikutnya dapat dikembangkan dengan membuat sistem untuk mengenali angka lebih dari 1 digit.
- Pada penelitian berikutnya dapat digunakan metode lain guna dapat dibandingkan dengan metode *Backpropagation*.

DAFTAR PUSTAKA

Achmad F.A. 2009.Perangkat Lunak Pengkonversi Teks Tulisan Tangan Menjadi Teks Digital. Skripsi. Surabaya. Institut Teknologi Sepuluh Nopember

Ahmad, A. M., Ismail, S., & Samaon, D. F. 2005. Recurrent Neural Network with Backpropagation through Time for Speech Recognition. International Symposium on Communications and Information Technologies. Oktober 26-29

Ardhianto, E., Munawaroh, S. & Prihandono, A. 2011. "Pengolahan Citra Digital untuk Identifikasi Ciri Sidik Jari Berbasis Minutiae". Jurnal Teknologi Informatika DINAMIK. 16 (1): 3

Cahyono, G.P. 2006. Sistem Pengenalan Barcode Mengguanakan Jaringan Syaraf Tiruan Learning Vector Quantization. FMIPA ITS

Erico, D.H, Lydia W.S. 2011. "Penerapan Jaringan Syaraf Tiruan Metode Propagasi Balik Dalam Pengenalan Tulisan Tangan Huruf Jepang Jenis Hiragana dan Katakana". Jurnal Informatika, Vol. 7, No.1

Gonzales, R. C., dkk. 2002. "Digital Image Processing Using Matlab". Prentice Hall, Upper Saddle River, NJ

Hermawan, A. 2006. Jaringan Syaraf Tiruan: Teori dan Aplikasi. Yogyakarta: ANDI

Iqbal, M. 2009. "Dasar Pengolahan Citra Menggunakan MATLAB". Departemen Ilmu dan Teknologi Kelautan ITB

Kusumadewi, S. 2003. "Artificial Intelligence (Teknik dan Aplikasinya)". Yogyakarta: Graha Ilmu

Prasetyo, E. 2011. "Pengolahan Citra Digital dan Aplikasinya Menggunakan

Matlab". Yogyakarta: ANDI

Pratiarso, A., dkk .2009. "Perbandingan Metode POC, Backpropagation, Coding Pada Pembacaan Plat Nomor Kendaraan Berbasis Image Processing". Politeknik Elektronika Negeri Surabaya.

Raheja, J.L, dan Kumar, U. 2010. "*Human Facial Expression Detection From Detected In Captured Image Using Propagation Neural Network*". *Internasional journal of computer science and information technologi* (IJCSIT). 2 (1): 7

Saeed Al-Mansoori Int.2015."*Intelligent Handwritten Digit Recognition using Artificial Neural*".ISSN:2248-9622, Vol. 5, Issue 5, (part 3)

Sari, Z. W. 2010. "Pengenalan Pola Golongan Darah Menggunakan Jaringan Saraf Tiruan Backpropagation". Skripsi. Malang: Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri (UIN) Malang

Siang, J.J. 2005. "Jaringan Syaraf Tiruan dan Pemrogramannya Menggunakan MATLAB". Yogyakarta: ANDI

Wuryandari, M.D. dan Afrianto,I. 2012. Perbandingan Metode Jaringan Syaraf Tiruan Backpropagation Dan Learning Vector Quantization Pada Pengenalan Wajah. Jurnal Komputer dan Informatika (KOMPUTA). 1 (1): 1

Zakson, A.M, 2013."Jaringan Syaraf Tiruan Algoritma *Backpropagation* untuk Menentukan Kelulusan Sidang Skripsi". ISSN:2301-9425

LAMPIRAN-LAMPIRAN

Coding Pembuatan dan Pelatihan Jaringan

```
latih=[x1a;x1b;x1c;x1d;x1e;x2a;x2b;x2c;x2d;x2e;x3a;x3b;x3c;x3d;x3e;x4a;x4b;x4
c;x4d;x4e;x5a;x5b;x5c;x5d;x5e;x6a;x6b;x6c;x6d;x6e;x7a;x7b;x7c;x7d;x7e;x8a;x8b
;x8c;x8d;x8e;x9a;x9b;x9c;x9d;x9e;x10a;x10b;x10c;x10d;x10e];

target=[1;1;1;1;1;2;2;2;2;2;3;3;3;3;3;4;4;4;4;4;5;5;5;5;5;6;6;6;6;6;7;7;7;7;7
;8;8;8;8;8;9;9;9;9;9;10;10;10;10;10];

newdatalatih = transpose(latih);
newtarget = transpose(target);
edit newff
nntool
```

Coding Pembuatan Sistem Pengenalan

```
function varargout = charGUI2(varargin)
% CHARGUI2 M-file for charGUI2.fig
%   CHARGUI2, by itself, creates a new CHARGUI2 or raises the existing
%   singleton*.
%
%   H = CHARGUI2 returns the handle to a new CHARGUI2 or the handle to
%   the existing singleton*.
%
%   CHARGUI2('CALLBACK',hObject,eventData,handles,...) calls the local
%   function named CALLBACK in CHARGUI2.M with the given input arguments.
%
%   CHARGUI2('Property','Value',...) creates a new CHARGUI2 or raises the
%   existing singleton*.  Starting from the left, property value pairs are
%   applied to the GUI before charGUI_OpeningFunction gets called.  An
%   unrecognized property name or invalid value makes property application
%   stop.  All inputs are passed to charGUI2_OpeningFcn via varargin.
%
%   *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%   instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
% Edit the above text to modify the response to help charGUI2
% Last Modified by GUIDE v2.5 19-Oct-2016 11:28:23
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @charGUI2_OpeningFcn, ...
                   'gui_OutputFcn',  @charGUI2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT



% --- Executes just before charGUI2 is made visible.
function charGUI2_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to charGUI2 (see VARARGIN)
load data; %MEMANGGIL JARINGAN
assignin('base','net',net);
% Choose default command line output for charGUI2
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes charGUI2 wait for user response (see UIRESUME)
% uiwait(handles.figure1);



% --- Outputs from this function are returned to the command line.
function varargout = charGUI2_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

%--------------------------------------------------------------------------
%MENGAMBIL GAMBAR
% --- Executes on button press in pbLoad.
function pbLoad_Callback(hObject, eventdata, handles)
% hObject    handle to pbLoad (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[filename, pathname] = uigetfile({'*.bmp';'*.jpg';'*.gif';'*.*'}, 'Pick an
Image File');
S = imread([pathname,filename]);
axes(handles.axes1);
imshow(S);

handles.S = S;
guidata(hObject, handles);
```

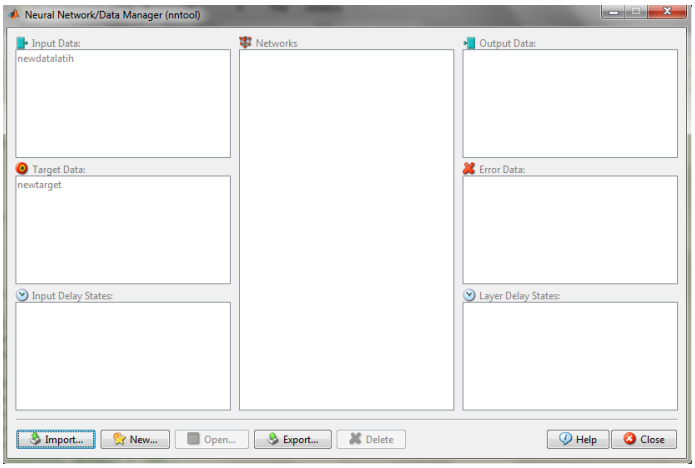
```
%--------------------------------------------------------------------
%SELECT
% --- Executes on button press in pbSelect.
function pbSelect_Callback(hObject, eventdata, handles)
% hObject    handle to pbSelect (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
S = handles.S;
axes(handles.axes1);

% Selection of location
if isfield(handles,'api')
    handles.api.delete();
    rmfield(handles,'api');
    rmfield(handles,'hRect');
    axes(handles.axes1);
    imshow(S);
end

axes(handles.axes1);
sz = size(S);
handles.hRect = imrect(gca,[round(sz(2)/2) round(sz(1)/2) 10 10]); % Select
object
handles.api = iptgetapi(handles.hRect);
guidata(hObject, handles);

%--------------------------------------------------------------------
%CROP
% --- Executes on button press in pbCrop.
function pbCrop_Callback(hObject, eventdata, handles)
% hObject    handle to pbCrop (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

handles.loc = handles.api.getPosition();
axes(handles.axes1);
S = handles.S;

handles.img_crop = imcrop(S,handles.loc);
axes(handles.axes2);
imshow(handles.img_crop);
guidata(hObject, handles);

%--------------------------------------------------------------------
%PREPROCESS
% --- Executes on button press in pbPreprocess.
function pbPreprocess_Callback(hObject, eventdata, handles)
% hObject    handle to pbPreprocess (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%RGB KE BINER
img_crop = handles.img_crop;
```

```
imgGray = rgb2gray(img_crop);
bw = im2bw(img_crop,graythresh(imgGray));
axes(handles.axes3);
imshow(bw);
handles.bw = bw;
bw2 = edu_imgcrop(bw);
axes(handles.axes4);
imshow(bw2);
handles.bw2 = bw2;

%TABEL NILAI PIXEL
a=uint8(bw);
k=size(a,2);
b=size(a,1);
c=k*b;
a=reshape(a,1,c);
b=(a');
myform = guidata(gcbo);
set(myform.tabel,'data',b);

bw2  = handles.bw2;
charvec = edu_imgresize(bw2);
axes(handles.axes5);
plotchar(charvec);
handles.charvec = charvec;
guidata(hObject, handles);

%--------------------------------------------------------------------
%EXTRAKSI CIRI
% --- Executes on button press in pbExtract.
function pbExtract_Callback(hObject, eventdata, handles)
% hObject    handle to pbExtract (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
guidata(hObject, handles);


%--------------------------------------------------------------------
%PENGENALAN CITRA
% --- Executes on button press in pbRecognize.
function pbRecognize_Callback(hObject, eventdata, handles)
% hObject    handle to pbRecognize (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
charvec = handles.charvec;
selected_net = get(handles.editNN,'string');
selected_net = evalin('base',selected_net);
result = sim(selected_net,charvec);
[val, num] = max(result);
set(handles.editResult, 'string',num);

% --- Executes on button press in pbNN.
function pbNN_Callback(hObject, eventdata, handles)
% hObject    handle to pbNN (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

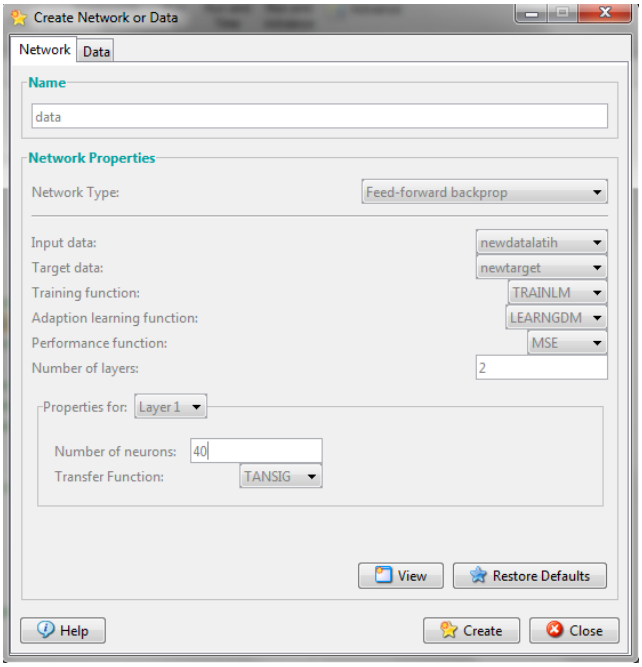
```
function editNN_Callback(hObject, eventdata, handles)
% hObject    handle to editNN (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editNN as text
%        str2double(get(hObject,'String')) returns contents of editNN as a
double

% --- Executes during object creation, after setting all properties.
function editNN_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editNN (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function editResult_Callback(hObject, eventdata, handles)
% hObject    handle to editResult (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editResult as text
%        str2double(get(hObject,'String')) returns contents of editResult as
a double

% --- Executes during object creation, after setting all properties.
function editResult_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editResult (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

Tampilan Proses Pembuatan Jaringan Pada Matlab

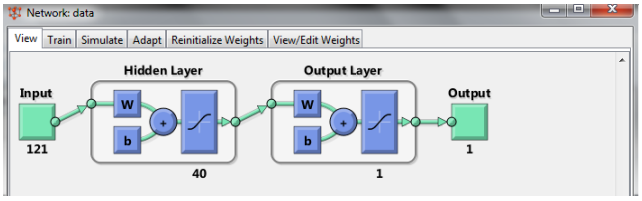
- Langkah 1



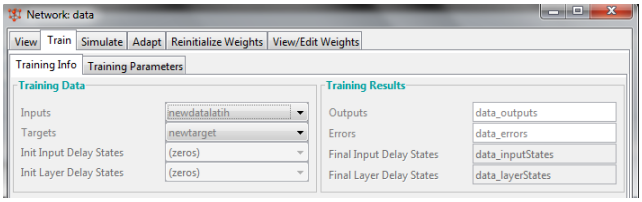
- Langkah 2



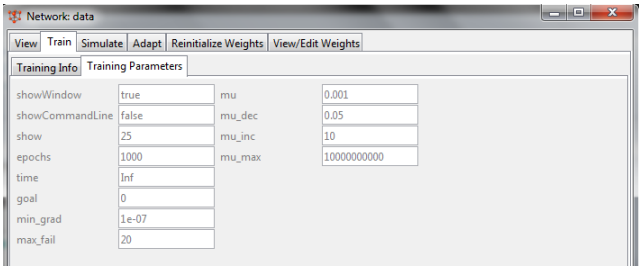
- Langkah 3



- Langkah 4



- Langkah 5



SURAT PERNYATAAN KEASLIAN SKRIPSI

Dengan ini saya yang bertanda tangan di bawah ini, mahasiswa Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Jakarta:

Nama	:	Alphien Andana
No. Registrasi	:	3125120206
Jurusan	:	Matematika
Program Studi	:	Matematika

Menyatakan bahwa skripsi ini yang saya buat dengan judul "**Pengenalan Citra Tulisan Tangan dengan Metode *Backpropagation***" adalah :

1. Dibuat dan diselesaikan oleh saya sendiri.
2. Bukan merupakan duplikat skripsi yang pernah dibuat oleh orang lain atau jiplakan karya tulis orang lain.

Pernyataan ini dibuat dengan sesungguhnya dan saya bersedia menanggung segala akibat yang timbul jika pernyataan saya tidak benar.

Jakarta, Januari 2017

Yang membuat pernyataan

Alphien Andana

DAFTAR RIWAYAT HIDUP

Alphien Andana Lahir di Batam, 12 Juni 1994. Anak pertama dari pasangan Bapak Alkadriman dan Ibu Sri Apiyati. Saat ini bertempat tinggal di Kavling Serpong Rt 01 Rw 04 No 19, Tangerang Selatan 15311.

No. Ponsel : 082112380321

Email : alphien55@gmail.com

Riwayat Pendidikan : Penulis mengawali pendidikan di TK Islam Al-Alif selama 1 tahun, dan kemudian melanjutkan pendidikan di SD Negeri Serpong 2 pada tahun 2000 - 2006. Setelah itu, penulis melanjutkan ke SMP Negeri 1 Serpong hingga tahun 2009. Kemudian kembali melanjutkan ke SMK Negeri 4 Tangerang dengan mengambil jurusan Rekayasa Perangkat Lunak dan lulus tahun 2012. Di Tahun yang sama penulis melanjutkan ke Universitas Negeri Jakarta (UNJ), jurusan Matematika, melalui jalur SNMPT jalur undangan. Di awal tahun 2017 penulis telah memperoleh gelar Sarjana Sains untuk Jurusan Matematika, Program Studi Matematika, FMIPA, UNJ.

Riwayat Organisasi : Selama di bangku perkuliahan, penulis ikut serta dalam organisasi kemahasiswaaan. Dalam satu tahun pertama, penulis mendapat kepercayaan sebagai staff Departemen Informasi dan Komunikasi (INFOKOM) di BEM Jurusan Matematika.

Riwayat Pekerjaan : Penulis mulai menjadi pengajar matematika sejak tahun 2011 dengan mengajar les private untuk siswa SD dan SMP dan berlangsung sampai tahun 2016. Penulis memulai bisnis online dengan menjual produk kebutuhan wanita seperti dompet, tas, baju dan aksesoris pada tahun 2016.